

University of Warwick institutional repository: <http://go.warwick.ac.uk/wrap>

A Thesis Submitted for the Degree of PhD at the University of Warwick

<http://go.warwick.ac.uk/wrap/1193>

This thesis is made available online and is protected by original copyright.

Please scroll down to view the document itself.

Please refer to the repository record for this item for information to help you to cite it. Our policy information is available from the repository home page.

**Emergence in the Security of Protocols for Mobile
Ad-Hoc Networks**

by

Ioannis Pavlosoglou

**A thesis submitted in partial fulfilment of the requirements for the degree
of Doctor of Philosophy in Engineering**

University of Warwick, Department of Engineering

July 2005

Table of Contents

- List of illustrations v
- List of tables..... viii
- Acknowledgements ix
- Abstract x
- Abbreviations xi
- Chapter 1: Wireless networking environments 1
 - 1.1 Overview 2
 - 1.2 Infrastructureless networking..... 4
 - 1.3 The mobile ad hoc networking environment 5
 - 1.4 Issues of security 6
 - 1.5 Problem description and methodology 8
 - 1.5.1 Protocol characterisation..... 9
 - 1.5.2 Emergence, a first encounter 10
 - 1.5.3 Model description 13
 - 1.6 Original contributions..... 15
 - 1.7 Disposition..... 18
- Chapter 2: Wireless routing protocols 19
 - 2.1 Classification 21
 - 2.2 Dynamic Source Routing (DSR) 24
 - 2.2.1 Route discovery and route maintenance 24
 - 2.2.2 Protocol characteristics 26
 - 2.3 Destination Sequenced Distance Vector (DSDV) 27
 - 2.3.1 Description and origin 28
 - 2.3.2 Protocol characteristics 29
 - 2.4 Ad hoc On Demand Distance Vector (AODV) 30
 - 2.4.1 Route tables, route discovery and route maintenance 30
 - 2.4.2 Protocol characteristics 33
 - 2.5 Temporally Ordered Routing Algorithm (TORA) 34
 - 2.5.1 Route creation, route maintenance and route erasure 34
 - 2.5.2 Protocol characteristics 36
 - 2.6 Zone Routing Protocol (ZRP) 37
 - 2.6.1 The concept of routing zones 37
 - 2.6.2 Protocol characteristics 40
 - 2.7 Cluster Gateway Switch Routing (CGSR) 41
 - 2.7.1 The interoperation of clusters 41
 - 2.7.2 Protocol characteristics 43
 - 2.8 AntNet 44
 - 2.8.1 Understanding ant colonies 45
 - 2.8.2 Ant based routing 47
 - 2.8.3 Similar protocol designs 49
 - 2.9 Methods of comparison 51
 - 2.9.1 Protocol complexity 51
 - 2.9.2 Single based characteristics 54
 - 2.9.3 Typical characteristics 55

2.10 Conclusions	58
Chapter 3: Security in wireless routing protocols	60
3.1 The elements of security in routing protocols	62
3.1.1 The computer element	64
3.1.2 The human element	65
3.2 Consequences of compromise	67
3.3 Classification of a malicious attack	68
3.3.1 Attack definitions	69
3.3.2 Classifying adversarial behaviour	74
3.3.3 Active attacker hierarchy	79
3.4 Protocol attacks	80
3.5 Security mechanisms and defence techniques	84
3.5.1 Cryptography and authentication	85
3.5.1.1 Basic terminology	86
3.5.1.2 Symmetric key cryptography	88
3.5.1.3 Public key cryptography	90
3.5.1.4 Digital signature algorithms	92
3.5.1.5 Hash functions and Message Authentication Codes	94
3.5.2 Intrusion detection	96
3.5.2.1 Solution tangents	97
3.5.2.2 Protocol correctness	99
3.6 Protocols incorporating security	101
3.6.1 The Secure Routing Protocol	101
3.6.2 Ariadne	103
3.7 Conclusions	105
Chapter 4: Emergence	107
4.1 Introduction	108
4.1.1 Self-Organisation	108
4.1.2 Randomness, Complexity and Chaos	111
4.1.3 Swarm Intelligence and Stigmergy	113
4.2 Defining emergent behaviour	114
4.3 A novel system: Pandemonium	117
4.4 Adaptive systems modelling	120
4.5 Limitations of operation	124
4.6 Design pitfalls	126
4.7 Conclusions	128
Chapter 5: The design of the Swarm simulator system	131
5.1 Implementation decisions	132
5.2 Requirements specification: Problem description	134
5.2.1 Describing the MANET	134
5.2.2 Describing the adaptive pandemonium	136
5.2.3 Describing user interactions	140
5.3 Requirements analysis: Use case model	143
5.4 Static modelling of the problem domain	148
5.4.1 Modelling the network	149
5.4.2 Modelling the scenario	151
5.4.3 Modelling the protocol	155

5.5 Object structuring of sub-systems	165
5.6 Dynamic modelling.....	168
5.7 Swarm: An adaptive protocol selection system	173
5.8 A Summary	175
Chapter 6: Secure protocol designs.....	177
6.1 Solutions to the black hole problem.....	178
6.1.1 Network and scenario configurations	178
6.1.2 Results obtained for packet drops	180
6.1.3 Feasibility analysis and discussion.....	189
6.2 Utilising cryptographic primitives within a protocol.....	193
6.2.1 Network and scenario configurations	194
6.2.2 Results obtained	196
6.2.3 Feasibility analysis and discussion.....	201
6.3 Combining protocol characteristics with cryptography	204
6.3.1 Network and scenario configurations	205
6.3.2 Results obtained	206
6.3.3 Feasibility analysis and discussion.....	211
6.4 Conclusions	215
Chapter 7: Conclusions and future directions	217
7.1 Overview	218
7.2 The applicability of Protocol Y	220
7.3 Future directions	222
7.3.1 Improvements on the secure protocol package	222
7.3.2 A second generation pandemonium	223
Appendix A: List of publications.....	225
References	226

List of illustrations

Figure 1.1: A visualisation of a MANET, consisting of a number of nodes that are made up from merging and fragmenting subnets.....	5
Figure 1.2: The compartmental model description in its constituent layers for the system used in automating the design of wireless routing protocols.....	14
Figure 2.1: The a) transmission of a DSR RREQ packet from source S, targeting destination D; b) propagation of the route reply from D back to S.....	25
Figure 2.2: The a) transmission of an AODV RREQ packet from source S, targeting destination D; b) propagation of the route reply from D back to S.....	31
Figure 2.3: The propagation of a route request in TORA from source to destination, based on the respective “height” of each MN. The resulting graph represents a DAG.....	35
Figure 2.4: The functionality of ZRP, separating the network into respective zones for source node S and destination node D.....	38
Figure 2.5: The process of routing packets from the source MN 1, to the destination MN 8 via cluster heads, utilising CGSR.....	42
Figure 3.1: The sub-elements within the computer element of ad hoc network systems security.....	64
Figure 3.2: The decision making process within the human element and the main factors affecting it.....	66
Figure 3.3: The categorisation of rankings within the dimension involving the collusion of power of adversaries.....	77
Figure 3.4: Attacker hierarchy using the active – n – m model, graded from the least (bottom) to the most (top) effective with regards to the consequences from a compromise.....	80
Figure 3.5: The principle of symmetric key cryptography and how it overcomes issues of passive eavesdropping.....	88
Figure 3.6: The principle of public key cryptography and how it overcomes issues of exchanging a secret key.....	91
Figure 3.7: The principle of digital signatures through the use of public key cryptography.....	93
Figure 3.8: The categorisation of message integrity functions into hash functions and message authentication codes, depending on the use of a secret key.....	95
Figure 3.9: A (4,2) threshold cryptographic scheme incorporating digital signatures; MNs 1 and 4 sign the message m with their partial key, thus allowing for the corresponding signature to be generated.....	98
Figure 4.1: An instance in the simulation of a 2D sand pile model; the height of the sand pile in different locations (left) of the plane is representing by different colours. The right square shows the locations that toppled after the addition of the last grain of sand.....	109
Figure 4.2: A system of finite interacting structures, under the observation of an observer P. Interactions are represented as links between structures.....	116
Figure 4.3: Selfridge’s Pandemonium model, as depicted in [216]. The interacting layers can be seen in boxes (excluding the image demon) attempting to make a decision on the character which they are identifying.....	118

Figure 4.4: The abstract layered hierarchy within Selfridge's Pandemonium model, exemplifying the interactions between layer 2 and layer 3. In order to be uniquely identifiable, demons of each layer carry a unique tag, shown as Dat67, Inf71, Des08, etc.....	119
Figure 4.5: An abstract description of an adaptive system selecting iterative structures from an environment E, with an adaptive plan τ	123
Figure 4.6: A circular mill of army ants, stuck in an infinite loop circle, trailing on the pheromone patterns of other ants.....	128
Figure 5.1: A two dimensional network schematic with 15 nodes, illustrating their corresponding links between them, as well as the broadcasting radius of each MN.....	134
Figure 5.2: Layer 3 within the adaptive model, containing the sub-systems relating to the protocol description, the communication scenario and the output, each with their corresponding attributes.....	137
Figure 5.3: The labelling of the three fundamental entities, namely the environment, the adaptive plan and the measure of performance within the compartmental model description in its constituent layers.....	139
Figure 5.4: A schematic view of the GUI control and display panel, allowing the necessary user interactions.....	141
Figure 5.5: Adaptive Protocol Design System Use Case Model.....	144
Figure 5.6: Conceptual static model for the network, corresponding to the physical classes that constitute a MANET.....	150
Figure 5.7: Conceptual static model for the scenario, corresponding to any communication scenario occurring within a MANET.....	151
Figure 5.8: Conceptual static model for the protocol, illustrating the use of the corresponding cryptographic classes.....	164
Figure 5.9: A system overview in UML of the package dependencies within Swarm.....	166
Figure 5.10: The UML sequence diagram detailing the creation of a communication scenario within Swarm.....	169
Figure 5.11: The UML sequence diagram detailing the adaptive selection process of a communication protocol.....	170
Figure 5.12: The UML class diagrams for the three adaptive classes, namely Demon, Measure and Plan.....	171
Figure 5.13: The GUI of Swarm consisting of four separate panels.....	173
Figure 5.14: The complete class and package hierarchy within Swarm.....	174
Figure 6.1: Results obtained for the characteristic pair of LSR versus DVR, following 100 simulations of the same black hole scenario.....	181
Figure 6.2: Results obtained for the characteristic pair of table-driven versus on-demand routing, following 100 simulations of the same black hole scenario	181
Figure 6.3: Results obtained for the characteristic pair of periodic versus event driven updates, following 100 simulations of the same black hole scenario.....	182
Figure 6.4: Results obtained for the characteristic pair of a flat versus a hierarchical structure, following 100 simulations of the same scenario.....	182
Figure 6.5: Results obtained for the characteristic pair of decentralised versus distributed route computation, following 100 simulations of the same scenario.	183
Figure 6.6: Results obtained for the characteristic pair of hop-by-hop versus source routing, following 100 simulations of the same scenario.....	183

Figure 6.7: Results obtained for the characteristic pair of single versus multiple paths, following 100 simulations of the same scenario.....	184
Figure 6.8: The protocol specification, as defined through its characteristic pairs, having being identified as the fittest candidate to tackle the black hole problem.....	185
Figure 6.9: The packet delivery ratio for Protocol X and also DSR versus the pause time.....	187
Figure 6.10: The packet overhead for Protocol X and also DSR versus the pause time.....	187
Figure 6.11: The byte overhead for Protocol X and also DSR versus the pause time.....	188
Figure 6.12: The median latency for Protocol X and also DSR versus the pause time.....	189
Figure 6.13: The selection range used for each characteristic pair.....	190
Figure 6.14: The four different protocol methods, as selected by Swarm, for the transmission of control packets within DSDV.....	196
Figure 6.15: The packet delivery ratio for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.....	198
Figure 6.16: The packet overhead for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.....	199
Figure 6.17: The byte overhead for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.....	200
Figure 6.18: The median latency for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.....	200
Figure 6.19: The three different protocol methods, as selected by Swarm, for sending, receiving and forwarding control packets within Protocol X.....	207
Figure 6.20: The packet delivery ratio for Protocol X and also Protocol Y versus the pause time.....	209
Figure 6.21: The packet overhead for Protocol X and also Protocol Y versus the pause time.....	209
Figure 6.22: The byte overhead for Protocol X and also Protocol Y versus the pause time.....	210
Figure 6.23: The median latency for Protocol X and also Protocol Y versus the pause time.....	211

List of tables

Table 1.1: Brief description of the currently available wireless standards.....3

Table 1.2: The three attributes that constitute data security in communications..... 7

Table 1.3: Three systems reaching a certain threshold of complexity where a new identity emerges that cannot be reduced to the sum of its parts..... 12

Table 2.1: The fields that each entry within the routing table of AODV must carry..... 30

Table 2.2: Comparison of the complexity characteristics of the protocols reviewed..... 53

Table 3.1: The rankings within the dimension involving the malignancy of an attack, graded from the least (top) to the most (bottom) effective with regards to the consequences from a compromise..... 76

Table 3.2: The rankings within the dimension involving the adaptivity and intelligence of an adversary, graded from the least (top) to the most (bottom) effective with regards to the consequences from a compromise..... 78

Table 5.1: Detailed description of the CreateScenario use case..... 144

Table 5.2: Detailed description of the CreateProtocol use case..... 145

Table 5.3: Detailed description of the FormatOutput use case..... 146

Table 5.4: Detailed description of the ControlSimulation use case..... 147

Table 5.5: Detailed description of the QueryOutput use case..... 147

Table 5.6: The definition of the malicious level attribute, as a five digit number for every TimeMalicious class..... 154

Acknowledgements

The dividend of the work seen here within represents a collection of attributes for the purpose of science on three distinct but highly correlated levels. Firstly, towards both my first and co-supervisor within the School; this thesis would not be in its current form if it was not for the relation we have had throughout my time in the University of Warwick.

Second, the research community in this field area, actively seeking to improve, learn and contrast on the evolution of outputs studied. For, you can easily identify the passion placed within their work and distinguish the efforts that help motivate people new in the subject area.

Lastly, the students that I have been actively involved with; for it is the nature of their questions that often provide solution paths and answers, yielding aspects that have previously gone unnoticed. They are my true mentors.

Abstract

This thesis is concerned with the study of secure wireless routing protocols, which have been deployed for the purpose of exchanging information in an ad-hoc networking environment.

A discrete event simulator is developed, utilising an adaptive systems modelling approach and emergence that aims to assess networking protocols in the presence of adversarial behaviour. The model is used in conjunction with the characteristics that routing protocols have and also a number of cryptographic primitives that can be deployed in order to safeguard the information being exchanged. It is shown that both adversarial behaviour, as well as protocol descriptions can be described in a way that allows for them to be treated as input on the machine level.

Within the system, the output generated selects the fittest protocol design capable of withstanding one or more particular type of attacks. As a result, a number of new and improved protocol specifications are presented and benchmarked against conventional metrics, such as throughput, latency and delivery criteria. From this process, an architecture for designing wireless routing protocols based on a number of security criteria is presented, whereupon the decision of using particular characteristics in a specification has been passed onto the machine level.

Abbreviations

ACO	Ant Colony Optimisation
AES	Advanced Encryption Standard
AODV	Ad hoc On Demand Distance Vector
ARA	Ant-Colony based Routing
BRP	Bordercast Routing Protocol
CBR	Constant Bit Rate
CGSR	Clustered Gateway Switch Routing
DAG	Directed Acyclic Graph
DARPA	Defence Advanced Research Projects Agency
DDOS	Distributed Denial Of Service
DES	Data Encryption Standard
DOS	Denial Of Service (a.k.a. DoS)
DSA	Digital Signature Algorithm
DSDV	Destination Sequenced Distance Vector
DSR	Dynamic Source Routing
DSS	Digital Signature Standard
DSSS	Direct Sequence Spread Spectrum
DVR	Distance Vector Routing
FHSS	Frequency Hoping Spread Spectrum
FIFO	First-In-First-Out
FILO	First-In-Last-Out
GPS	Global Positioning System
GUI	Graphical User Interface

IARP	Intrazone Routing Protocol
IDS	Intrusion Detection System
IDS	Intrusion Detection System
IEEE	Institute of Electronic and Electrical Engineers
IERP	Interzone Routing Protocol
IETF	Internet Engineering Task Force
IMEP	Internet MANET Encapsulation Protocol
IP	Internet Protocol
IrDA	Infrared Data Association
LCC	Least Cluster Change
LOS	Line Of Sight
LSR	Link State Routing
MAC	Message Authentication Code
MANET	Mobile Ad hoc Network
MN	Mobile Node
NIST	National Institute of Standards and Technology
NPDU	Network Protocol Data Unit
OMG	Object Management Group
OO	Object-Oriented
PDR	Packet Delivery Ratio
RREQ	Route Request (Packet)
SA	Security Association
SHA	Secure Hash Algorithm
SI	Swarm Intelligence
SOC	Self-Organised Criticality

TORA	Temporally Ordered Routing Algorithm
UML	Unified Modelling Language
UTM	Universal Turing Machine
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
ZRP	Zone Routing Protocol

It would have to be,

To my parents,

For all the simple logic that they carry

Wireless networking environments

1.1	Overview
1.2	Infrastructureless networking
1.3	The mobile ad hoc networking environment
1.4	Issues of security
1.5	Problem description and methodology
1.5.1	Protocol characterisation
1.5.2	Emergence, a first encounter
1.5.3	Model description
1.6	Original contributions
1.7	Disposition

This chapter presents a general review of the thesis, introducing the concept of infrastructureless networking as well as one of its realisations, the mobile ad hoc network. Furthermore, issues of security are examined and the reasoning behind the amplification of routing security in ad hoc networking environments is presented. The main section focuses on the methodology followed in tackling issues of routing security, presenting the unique contribution of emergence in the modelling process. Finally, the original contributions of this thesis are presented, followed by a brief chapter outline and the dependencies between them.

1.1 Overview

The continuous emergence of mobile handheld devices in daily life has necessitated the use of wireless techniques for the exchange of information between them. In this rising demand for service availability to non-fixed users with varying locations, achieving the same level of connectivity but ‘without the cat’¹ has proven to be a substantial challenge. Further to this, the dynamic structure of any wireless network formed, has yielded an overwhelming potential growing beyond the premeditated capabilities of equivalent wire based connections. Consequently, the design of protocol implementations for ‘no-cat’ networks, with all the difficulties they might pose, is transpiring as a new field of unique prerequisites and conditions, verifying once again that the message is the medium.

In attempting to provide for this surfacing technology and the potential that it holds, we must question the driving forces related to it. In the brief history of telecommunications network evolution, three forces have been instigating the resulting architectures [1]: traffic growth, development of new services and advances in technology. From these, traffic growth relates to the continuous increase of information exchange, embracing speeds per unit of bandwidth, as well as the handling of connections to different users. The development of new

¹ *“You see, wire telegraph is a kind of a very, very long cat. You pull his tail in New York and his head is meowing in Los Angeles. Do you understand this? And radio operates exactly in the same way: You send signals here and receive them there. The only difference is that there is no cat.”* Albert Einstein [1879 – 1955]

services, tightly related to advances in technology, relates to ways and methods of manipulating information and the means by which it is being exchanged. Despite of the above not being independent of each other, each one of these aspects has helped shape telecommunication networks in different ways.

Table 1.1: Brief description of the currently available wireless standards.

Standard	Description	Usage
IEEE 802.11 [2]	Can use either Direct Sequence Spread Spectrum (DSSS) or Frequency Hopping Spread Spectrum (FHSS) in the 2.4 GHz band, as well as Pulse Position Modulation for Infrared (IR) transmission. Could achieve data rates of up to 11 Mbps using DSSS.	Wireless Local Area Networks (WLANs). Replacing the current Ethernet network card in campus/office multi-user computer environments.
Bluetooth [3]	Based on the 802.11 standard, it is an RF specification operating in the 2.45 GHz band. It uses frequency hopping of about 1600 hops/sec changing 79 freq. at 1 MHz intervals and is designed for short-range communications of up to 10 m. Presently, it offers a data rate of approx. 1Mbps.	In house appliances, ranging from printers to cordless phones. Also popular with mobile phones where the user doesn't need to be in contact with the handset, thus reducing radiation hazards.
IrDA [4]	Two standards are available: The IrDA Data standard, recommended for high speed, line of sight (LOS) data transfer of 4 Mbps and the IrDA Control standard recommended for in-door cordless PC peripherals.	IrDA data can be found in most handheld devices, such as mobile phones, IrDA control is commonly used for wireless keyboards, etc.

The applicability of these three driving forces to no-cat networks has created a plane well outside the one dimensional aptitude of exchanging raw bits. The turn of this century saw a number of developing standards with the ability to deliver such connectivity. Table 1.1 summarises the currently available, hinting on their environment of operation. Having altered the medium of communication from wired to wireless, the questions that surface relate more and more to the limitations of our means to manipulate information in a

multiplicity of dynamic users with varying locations, than to the static communication environments that we have been utilising all these years.

1.2 Infrastructureless networking

In the realm of wireless communications, two system models have been proposed weighing their distinction upon the presence of a proxy within the nodes wanting to exchange information. The first approach, based on existing cellular infrastructures, involves a fixed backbone wireless system, where the network consists of a number of mobile nodes, as well as a number of fewer but more powerful fixed nodes. Such a system requires the presence of a permanent infrastructure that is hardwired using landlines. Some of the problems [5] in network realisations of this type (apart from the obvious need for a wired connection in order to establish a wireless channel) relate to the handoff between different devices in the backbone, as well as the smooth transition between nodes of the backbone, without noticeable delay or packet loss.

The second, shifting away from cellular operation, is to form a network on demand and in an ad hoc fashion, where all users and devices are not only clients in the network but are also willing to forward data packets on behalf of other nodes, in an attempt for them to be delivered from source to destination. This type of system model, often referred to as infrastructureless networking [6] allows for mobile nodes to dynamically establish routing between them and (dependant on the limitations their broadcasting range) with other nodes in a hop-by-hop fashion as seen in figure 1.1.

Between the two system models, despite that the former, due to the static element of the backbone capacitates a lot more power and hence offers a larger

broadcasting range, ad hoc networking offers a number of advantages spawning from the on demand setup that it subsists. An example of this is an environment that holds a damaged communication infrastructure such as a battlefield or a natural disaster recovery area. In such a situation, ad hoc networking is the ideal candidate for fast deployment of a communication channel among the users present.

1.3 The Mobile Ad Hoc Network

Based on the description of infrastructureless networking, comes the concept of a Mobile Ad Hoc Network (MANET). A MANET consists of a number of Mobile Nodes (MNs) that have no connectivity to backbone hosts and therefore do not require any management with respect to location or handover [7].

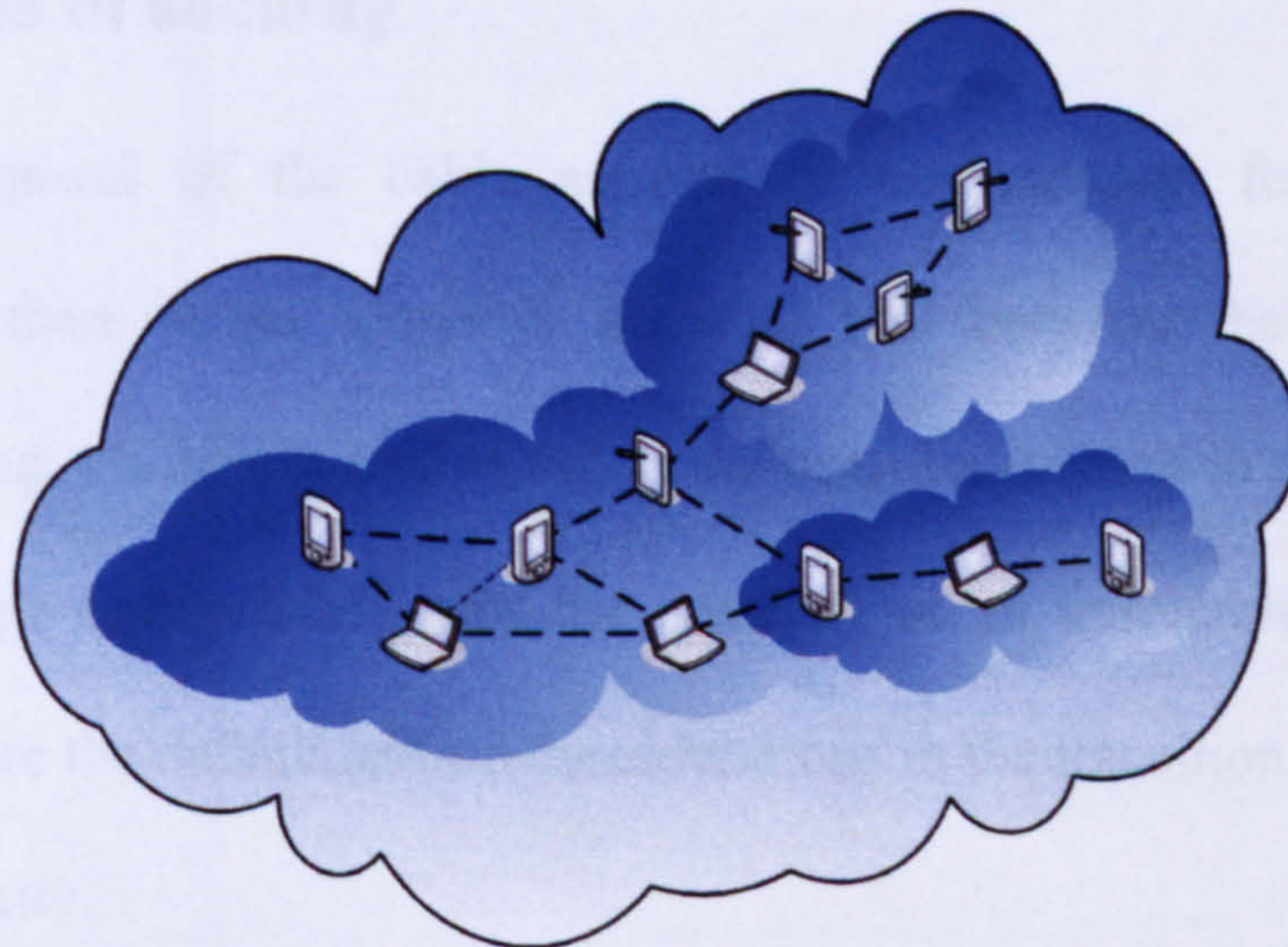


Figure 1.1: A visualisation of a MANET, consisting of a number of nodes that are made up from merging and fragmenting subnets.

Each MN acts on the assumption that no centralised administration is present within the network and relies on neighbouring MNs for the transmission and reception of information. As a result, routes are established depending on the location and broadcast range of every MN. Further to this, each MN acts as a router, forwarding traffic between nodes.

Ad hoc networks can also exist within a hybrid environment hosting a backbone infrastructure where only a few nodes have connectivity to the fixed link points. In either case, a MANET consists of a number of subnets or partitions where, as illustrated in figure 1.1, certain nodes act as gateways to other subnets within the networking environment. This takes place in a dynamic way, with subnets merging and fragmenting in real time depending on the mobility of individual or groups of MNs. Consequently, a MANET is an environment necessitating high cooperation for any data exchange to take place. This, adding to the open exchange of information lacking any cats, subsidises the need for greater protection in the communication exchange process.

1.4 Issues of security

With the removal of the cable as part of our medium for exchanging information, there is no' longer a physical link between the two devices communicating; all information is out in the open for anyone to get their hands on. Thus, high susceptibility plus the presence of an unreliable transmission medium [8] are the default add-on considerations in the transition from wired to wireless security.

The advantages of an on demand setup in MANETs add to the disadvantages of wireless security [9, 10]. Information is not only being exchanged in an open and more unreliable medium, but also in an environment of continuously changing topology, where traffic has to pass through other nodes for it to reach its final destination.

In an attempt to define the components of it, wireless data security can be seen as a superset of three attributes [11], all of which need to be maintained within a secure system. These can be seen in table 1.2 followed by a brief description as well as the question of compromise to see if each attribute has been breached. Such an approach focuses on the bits or data being exchanged and not on the requirements of the system for it to be labelled as secure. In any case, a secure system should have the ability of maintaining each of the attributes of table 1.2

Table 1.2: The three attributes that constitute data security in communications.

Attribute	Description	Question of compromise
Data integrity	Users having the ability to modify or delete data for which they do not have such privileges to do so	Has the data been improperly tampered with?
Data secrecy	Users having the ability to view data for which they do not have audit permissions to do so	Has the data been improperly disclosed?
Data Availability	Users not having the ability to access data that they have ownership of, or has been sent to them. Denial-of-Service (DOS) attacks are the most common threat to availability	Can I access my data when I want to?

With the escalation of wireless security in an environment hosting a MANET, each of the above attributes also becomes harder to maintain. Regarding data integrity, any user can broadcast information in the absence of a cat, as well as modify data packets passing through them. With respect to data secrecy, information is no longer sent down a cable, but broadcasted within a specified range. Finally with respect to availability, any node within a MANET can refuse to forward incoming packets during the exchange of information causing a communication breakdown. These issues (that will be examined in greater detail

in chapters to come) form the basis of wireless security outlining the principal types of attack that can occur in MANETs.

1.5 Problem description and methodology

From the general issues of security and how they are amplified in MANETs, at the core of the problem lie the freedoms that individual nodes have in viewing, manipulating and forwarding incoming data. These, extending to all three attributes of table 1.2, define the concept of routing security.

As we have seen, the cooperation of nodes within such a networking environment is integral to its existence. Thus, the protocol describing the rules for exchanging information should facilitate for malicious behaviour from one or more MNs that are part of the network. If this were not the case, a node choosing to compromise the secrecy, integrity or availability of data would cause the collapse of any information exchange. Consequently, in order to allow for the advantages that a MANET has to offer, we need to secure the communication process as that is defined in the routing protocol being utilised.

Due to the dynamic characteristics of MANETs, the protocol deployed would have to facilitate for malicious intent in a number of different situations and phases of the communication exchange. As a result, our problem description focuses on malicious behaviour within ad hoc networks and the ability of the routing protocol utilised to notice, notify and isolate such a node or groups of nodes from the remaining network. This task, if seen in proportion to the number of available protocol implementations is not at all trivial.

1.5.1 Protocol characterisation

At present a number of protocols exist for the routing of data in ad hoc networks. These (as we shall see in chapter 2) fall in a variety of categories, weighing a mixture of advantages and disadvantages depending on the communication scenario under question.

In designing a protocol for a MANET, not only must we take into consideration the issues of routing security, but also allow for the advantages that such a network has to offer. The “3 Anys” – [12, 13] Any person, Anywhere and Any time, illustrate the level of connectivity that ad hoc networks aim to offer. Thus, bearing in mind the nature of infrastructureless networking, any protocol specification needs to have the ability to deal with a number of complex scenarios stemming from a constantly changing topology.

As a result, every protocol must be described through a number of global characteristics that can withstand the resulting local complexities presented in a dynamic information exchange. If this were not the case, we would have to premeditate every possible scenario that a protocol can encounter and include it in the specification. Clearly, following Ockham’s razor², this is not something that we favour, or, in most cases, can actually achieve.

² The medieval rule of parsimony, or principle of economy, frequently used by the philosopher William of Ockham, Surrey, England (1285-1349) came to be known as Ockham's razor: “*Entia non sunt multiplicanda praeter necessitatem* or *Numquam ponenda est pluralitas sine necessitate*”; (in free translation) It is vain to do with more that which can be done with less.

Following the need for simple rules to govern complex and dynamically changing events, yields a system that has the ability to adapt to new situations. This process (as revealed in the next section) shares a lot of common characteristics with adaptive modelling and emergence. The passage from global description criteria, such as a protocol, that prove worthy of local phenomena, such as the occurrence of a malicious node, forms the basis of the modelling approach followed throughout this work.

1.5.2 Emergence, a first encounter

Over the later half of the previous century, studies focusing on the observation of collective behaviour in natural systems produced a number of fascinating, yet analogous results. The plane that initially enabled researchers to umbrella the observed patterns under a single common denominator spawned from an almost contradictory model involving complexity and simplicity. Despite the fact that the behaviour of the entities involved could be narrowed down to a small set of simple rules, the system, as a whole, had the ability to tackle often successfully, quite complex problems. As this was first observed in the natural, the results often mirrored simple facts that we tend to take for granted such as the way in which ants or bees manage their resources in their scavenge for survival.

Today, this field of research named emergence (due to the similarities that it has with the same term as used by 19th century biologists to describe what happened when life arose from non-living matter 4 billion years ago [14]) finds applicability in both natural and artificial systems. Whether it has been a case of understanding the behaviour of ant colonies in their struggle for survival [15], or for consumer websites to offer hints based on a returning customer's purchase

history [16], this new *kind of science*³, even at this early stage, has been offering unique results through its simplistic perspective.

Since emergence, as a science, can be hosted in so many different subjects, an interpretation of what it stands for comes in many different flavours. Three of the most common such descriptions are included below:

- Bottom-up systems achieving the same functionality as top-down

Systems lacking a central administration authority, which are capable of functioning in the same manner as if one was present. The myth of an ant queen in specific ant colonies is the perfect example of such a natural system [18]

- A set of simple rules generating complex behaviour

Systems governed by a set of very simple rules, having a complex, non-chaotic behaviour. A number of examples of such artificial systems can be found in [17, 19], where the system defined is a cellular automaton

- . The sum of the parts being greater than the whole

A principle stemming from Gestalt theory [20], stating that a sufficiently complex system can experience properties that cannot be reduced to the constituent elements of the system. Hence the resulting structure can tackle problems considered outside its initial capacity as that is defined by the principle of superposition, widely used in physics. A number of examples of such systems can be seen in table 1.3 below.

³ Title used by Stephen Wolfram in his latest book [17] on cellular automata, which extensively describes the surfacing of complex system behaviour, spawning from a set of very simple rules governing the system.

The above three interpretations of such phenomena briefly summarise the distinct behaviour that allows us to interpret a system as one that hosts or experiences emergent behaviour. An important aspect that must not be neglected (that shall be revisited in chapter 4) lies in the necessity but not sufficiency of the condition for emergence to occur. Even though a living organism is made up of molecules, a collection of molecules does not necessarily give rise to a living organism. As intuitive as this illustration might be, it hints on a required set of conditions for a new system identity to surface.

Table 1.3: Three systems reaching a certain threshold of complexity where a new identity emerges that cannot be reduced to the sum of its parts.

Resulting identity of the system	Original constituent parts	
Living organisms	<i>Emerge from</i>	Systems of molecules
Mind, consciousness and intelligence	<i>Emerge from</i>	Systems of neurons
Organisations, societies, and cultures	<i>Emerge from</i>	Systems of individuals

From observations in the natural, as well as problem solving techniques applied in the artificial, we grow aware of the fact that emergence does not take place unless a problem requiring a solution is present. Despite the Darwinian aspects of the latter statement, it seems that emergence offers the opportunity to the system of almost learning an alternative solution technique, thus adapting to the problem at hand. It is this element of adaptation that we aim to exploit in the development of wireless routing protocols, constantly presented with the problem of a malicious attack within a dynamic environment.

1.5.3 Model description

The objective of this work is to deliver a functioning system, capable of automating the design process for routing protocols. As a result, our focus shifts a level up; from the improvement of a single design to a system that can facilitate for the requirements of MANETs and deliver working protocol implementations. The main tool enabling us to pass the design phase of this process to machine level is emergence.

Developing a model that can adaptively learn to handle occurrences of malicious intent within a MANET and embed them on the resulting protocol utilised, falls outside conventional control theory [21, 22]. Normally, the described process would involve a mechanism that would improve the resulting process based on the feedback gained from both the good and bad aspects of a design.

Even though this fundamental aspect of feedback is one that should not be neglected, conventional methodology only allows for decisions to be made by taking into account information on the same level as that of the observed phenomenon. To draw a parallel to this, today the acting environment judge protocol implementations has been the corresponding community related to this field of research. Our objective is not to replace but to simplify this task, by further automating elements in the design process.

Extending on this assumption, our proposed model structure, seen in figure 1.2, contains a number of distinct layers. This categorisation comes as a result of the

information that each compartment within the model processes. On the top of the model lies the decision layer responsible for selecting a best fit for the protocol description though its interaction with neighbouring compartments. The reason why this compartment is placed at the very top of the hierarchy is because ultimately all information is filtered through this layer to a decision regarding the design. In choosing to label this compartment as a daemon, we take from Oliver Selfridge's Pandemonium described in section 4.2.

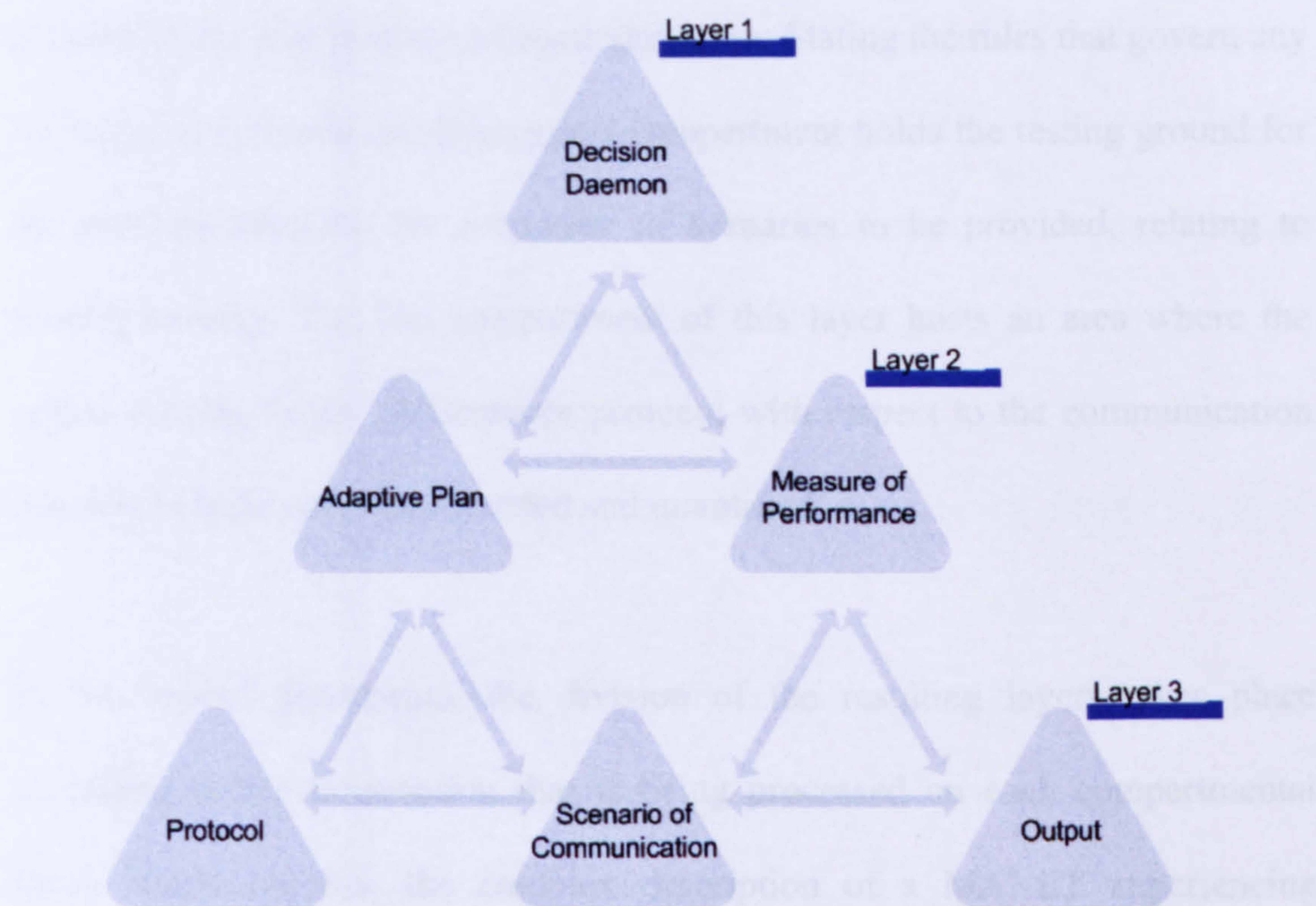


Figure 1.2: The compartmental model description in its constituent layers for the system used in automating the design of wireless routing protocols.

Layer 2 holds two compartments responsible for providing the information necessary to layer 1 for the decision making process. The first is the plan of adaptation stating how the next structure for evaluation should be selected based on the results obtained from the previous one. The second is a measure of performance providing means for assessing good or bad structures from the next layer. Between the two compartments there exists an unavoidable correlation

justifying each one's presence on the same layer. A strategy for adjusting the selection process cannot exist without quantifying the performance of the system and vice versa.

Finally, the last layer is constituted with the compartments that can be found in a model description of a control feedback process. From the three compartments found in this layer, the first one holds the protocol description as that would be defined for the sole purpose of communication: Stating the rules that govern any exchange of information. The second compartment holds the testing ground for the protocol allowing for a number of scenarios to be provided, relating to routing security. The last compartment of this layer hosts an area where the output relating to the performance protocol with respect to the communication scenario at hand could be recorded and quantified.

In this model description, the division of the resulting layers takes place according to the information that is being processed on each compartmental level. Based on this, the complex description of a MANET experiencing malicious activity can act as a test-bed for selecting a routing protocol and recursively improving on its characteristics, until it reaches a level of being able to overcome the imposed malicious intent.

1.6 Original contributions

The main contribution that will be exploited within this thesis focuses on the applicability of emergence in the design of wireless routing protocols. Current research tangents have only recently started to touch this field and to question the advantages that it has to offer. A prime example of this is the routing

protocol AntNet [23 – 25, 55 – 59] and the support that it has been winning as a routing technique in recent years. AntNet bases its operation in the simulation of pheromone patterns and the way in which they affect the decision making process within an ant colony. We will examine this protocol and the advantages that it has to offer in greater detail in section 2.8.

Despite this recent attempt to utilise a system hosting emergent characteristics within the rules underlying the resulting protocol structure, to the author's knowledge no attempt has been made in applying such methodology to the design phase for wireless routing protocols. It is this idea that forms the basis of originality within this work.

The contributions presented within this thesis can be broken down into three constituent parts.

- Firstly, the design of a system with the ability to in turn design, modify and improve on wireless routing protocols. The main toolbox enabling this process stems from emergence and practices involving adaptive modelling. The material used for this process is presented in chapter 2, 3 and 4, by examining routing protocols, routing security and emergent systems respectively.
- Secondly, the implementation within an object oriented language of a discrete event simulator capable of offering an environment that can develop routing protocols by correlating the rules of adaptive modelling

with the requirements of handling a malicious attack. This work along with the decision steps taken is presented in chapter 5.

- Thirdly, a number of protocol implementations, designed at machine level, which can tackle specific types of attack within a MANET are presented. For this, we initially focus on attacks that are passive, i.e. there are no alterations in the data traffic generated on the network and later look at more active types of attack. Chapter 6 focuses on turning the handle on the system created to obtain such protocol implementations through a set of known conditions and limitations.

Finally, in the process of achieving a level of contribution in each of the above three parts, a number of necessary and more detailed new elements have arisen.

These are:

- The derivation of an analogue protocol representation, based on a number of characteristic pairs (sub-section 5.4.3)
- The detailed description and categorisation of any malicious attack within a MANET as a five digit number (sub-section 5.4.2)
- A methodology for accessing a protocol security with respect to characteristic pairs (sub-sections 5.4.3 and 6.1.2)
- A methodology for accessing the way in which encryption techniques can be added to a protocol (sub-sections 5.4.3 and 6.2.2 and 6.3.2)

The combination of the above detailed structures with the goals presented herein forms the motivation behind this thesis.

1.7 Disposition

Following the contributions presented, this thesis has a tripartite structure, which is designed to get at many of the same themes from somewhat different angles. The first part consists of identifying in great detail the operation as well as failures of known protocol implementations, presented in chapters two and three. This carries an in depth analysis from a security perspective that helps us to identify the tools for avoiding communication pitfalls that can be exploited by an adversary.

The second part seen in chapter 4 is an extended analysis of emergent systems that includes the rules as well as design pitfalls of their deployment. Both previous chapters attempt to identify characteristics that can be used in the adaptive modelling process, which are inherent to protocol designs.

Thus, the final part combines the theory presented in chapters two, three and four, presenting the design of a simulator that has the ability to embrace the aspects of wireless protocol security and adaptive emergent designs. From this, stem a number of results that are presented in the penultimate chapter. This thesis concludes with a chapter dedicated to conclusions and further work that can be pursued in the cross-section of the correlated ideas that have been seen presented herein.

Wireless routing protocols

2.1	Classification
2.2	Dynamic Source Routing (DSR)
2.2.1	Route discovery and route maintenance
2.2.2	Protocol characteristics
2.3	Destination Sequenced Distance Vector (DSDV)
2.3.1	Description and origin
2.3.2	Protocol characteristics
2.4	Ad hoc On Demand Distance Vector (AODV)
2.4.1	Route tables, route discovery and route maintenance
2.4.2	Protocol characteristics
2.5	Temporally Ordered Routing Algorithm (TORA)
2.5.1	Route creation, route maintenance and route erasure
2.5.2	Protocol characteristics
2.6	Zone Routing Protocol (ZRP)
2.6.1	The concept of routing zones
2.6.2	Protocol characteristics
2.7	Cluster Gateway Switch Routing (CGSR)
2.7.1	The interoperation of clusters
2.7.2	Protocol characteristics

2.8	AntNet
2.8.1	Understanding ant colonies
2.8.2	Ant based routing
2.8.3	Similar protocol designs
2.9	Methods of comparison
2.9.1	Protocol complexity
2.9.2	Single based characteristics
2.9.3	Typical characteristics
2.10	Conclusions

This chapter details existing wireless routing protocol implementations, as well as the general categories into which these fall in. Selectively, benchmark protocol specifications are presented. Each one carries two sections, one describing the protocol functionality and another outlining its general characteristics as well as drawbacks. In greater detail, a review of DSR, DSDV, AODV, TORA, ZRP and CGSR are presented. Also, in the section on AntNet, protocols that base their operation on emergent observations from natural systems are described. This chapter concludes by reviewing the different methods of comparison for known protocol architectures, based on their complexity and also a set of typical and single based characteristics.

2.1 Classification

Following the Defence Advanced Research Projects Agency (DARPA) packet-radio project [26, 27] in the late 1970's, numerous protocols have been developed for ad hoc mobile networks. The attainment targets of such protocols from then until now have been confined to the typical limitations that such networks pose. These include [28] high power consumption, low bandwidth and high error rates. Up to the most recent specification, that being the IEEE 802.11 standard for Wireless Local Area Networks (WLANs) as it was first published in 1997 [2, 29] these limitations remain intact.

From the various routing protocols that have been proposed in the literature [28], a number of different criteria for classifying them exist. These are dependant on a number of parameters, such as the information being exchanged, the way in which routes are computed as well as the timing of the routing information. In synopsis, the criteria for the categorisation of routing protocols, reviewed in [30], are as follows.

- **Link state routing (LSR) vs. distance vector routing (DVR)**

In LSR any noticeable change in topology is immediately flooded to the entire network, while in DVR every node maintains a distance vector which is periodically exchanged with its neighbours. The trade-off is the routing overhead (in LSR) versus the slow convergence and routing loops (in DVR).

- **Table-driven routing vs. on-demand routing**

Table-driven routing, also referred to as proactive routing [31], involves the pre-computation of routes to all MNs present within the MANET. On

demand routing relates to the dynamic discovery of a route from the source MN to the destination upon wanting to exchange information. The trade-off is the attempt to keep an up to date view of all available routes in table-driven routing, versus a large latency at the beginning of every transmission in on-demand routing.

- **Periodical update vs. event-driven update**

Depending on the timing of routing information exchange on the network, periodical updates involve a slotted time period at which such information is transmitted. Event-driven updates are broadcast in the opportune moment where other traffic is travelling across the network. The trade-off in this case is the risk of consuming bandwidth from the communication channel at a constant rate (periodical update) versus not having any information with regards to routing unless a communication event takes place.

- **Flat structure vs. hierarchical structure**

In a flat structure all nodes within the MANET have the same responsibilities with respect to routing. In a hierarchical structure (also referred to as cluster-based routing) specific nodes act as a miniature central authority within a subset of MNs within the MANET, having the responsibility of maintaining routing and topology related information. The trade-off in this case is the long time that it takes to update routing information in flat structures, versus network partitioning and denial of service from a cluster head in a hierarchical structure, owing to the dynamically changing topology.

- **Decentralised computation vs. distributed computation**

Depending on the number of nodes that get involved in the computation of a route, decentralised computation assumes that each MN keeps track of the

complete network topology, thus allowing each node to independently compute the route to the destination. In distributed computing, the calculation of a route involves the collaboration of one or more nodes sharing the partial information that they have on the network. The trade-off in this case relates to attempting to keep information about the entire network in decentralised computation, versus gathering the necessary information for routing when required.

- **Source routing vs. hop-by-hop routing**

Determined by the amount of information that a packet contains concerning the route followed, source routing assumes that each packet holds all the information of intermediate nodes that the packet must go through to reach the specified destination. Hop-by-hop routing on the other hand, assumes that each packet carries only enough information about the next MN that the packet should go through. The trade-off in this case is bandwidth consumption in large networks for source routing, versus all nodes needing to maintain up to date information for hop-by-hop routing.

- **Single path vs. multiple paths**

A routing protocol can offer a single route or multiple routes to a specified destination. The trade-off is the bandwidth and memory requirements when transmitting information on multiple paths, versus the lack of a route recovery process in the event of a link failure for a single path.

Based on the above classifications, a number of protocols exist, embracing specific attributes. Selectively, we choose to focus on particular examples that present landmarks for wireless routing, in the sections that follow.

2.2 Dynamic Source Routing (DSR)

Starting with on demand routing protocols, Dynamic Source Routing (DSR) [32, 33, 34 35] belongs to the class of reactive protocols that allow nodes to dynamically discover a new route to the destination using multiple packet hops. Each packet sent carries in its header an ordered list of all the nodes that it has to pass to reach the destination. Consequently, the protocol offers two modes of operation, one for discovering routes within the network and one for maintaining an up to date view on a per node basis.

2.2.1 Route discovery and route maintenance

From the specification of DSR [33], a main focus is route discovery. When a MN, say S, wishes to transmit information to another MN, say D, it first looks at the stored information that it has on known routes. All nodes using DSR are required to store route caches containing the source routes to destination nodes they are aware of. Provided that S does not have an unexpired route [35] to the destination it will transmit a Route Request (RREQ) packet to all its neighbours. The objective of this process is to establish a route record that will be placed in the route cache of the MN for future reference.

Every node receiving this RREQ packet searches through its route cache to see if it has an unexpired route to D. If no route exists, the node is obliged to forward the RREQ packet further, adding its own address to the recorded hop sequence. This process (illustrated in figure 2.1a) continues in this manner through the network until either the RREQ packet reaches D, or it reaches a

node offering a route to D. When this takes place, the RREQ packet is returned back to S, in one of many ways.

The simplest way of retrieving the RREQ is by reversing the hop record with the packet header, as seen in figure 2.1b. The disadvantage of this method is that the network topology might have changed during the propagation of the RREQ packet. In order to overcome this limitation, DSR checks the route cache of any intermediate node for a more up to date route [33, 35]. If one is found, it is used instead of the suggested path offered in the header field.

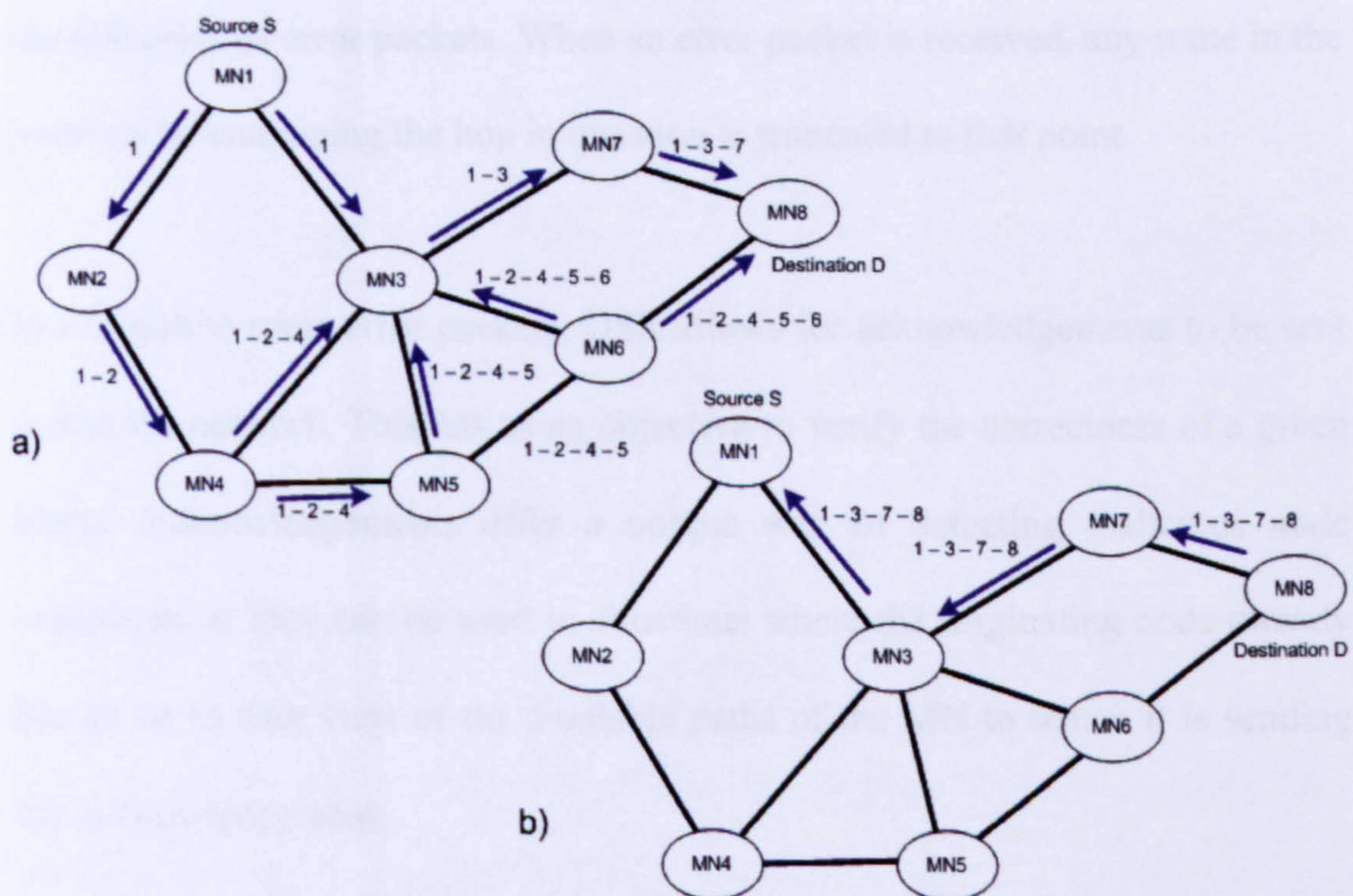


Figure 2.1: The a) transmission of a DSR RREQ packet from source S, targeting destination D; b) propagation of the route reply from D back to S.

DSR further expands its operation by allowing nodes to operate in promiscuous mode. This is done when a MN ceases to only pay attention to data packets targeted at it, but also scans the headers of all incoming packets, prior to forwarding them, for useful routing information. The use of this technique

increases the number of data routes stored in the route cache. Once a route has been found, the task of maintaining it begins.

Route maintenance has as an objective to keep an up to date view on network topology from every potential node S, with respect to node D. This is accomplished through the use of route error packets and acknowledgements. A route error packet is sent back to node S when another node being part of the transmission path from S to D, reports that it can no longer use that path. DSR does not examine the reason behind this failure nor does it provide a classification of error packets. When an error packet is received, any route in the route cache containing the hop in question is truncated to that point.

In addition to route error packets, DSR allows for acknowledgements to be sent across the network. This has as an objective to verify the correctness of a given route. Acknowledgements offer a unique way of detecting malicious node behaviour, as they can be used in situations where the originating node already has an up to date view of the available paths of the MN to which it is sending the acknowledgement.

2.2.2 Protocol characteristics

DSR capitalises on the advantages that source routing has to offer. Any MN part of the network is not required to have up to date routing information of the recipients they aim to reach. Furthermore, the establishment of new routes takes place on demand, without any periodic updates taking place at regular intervals.

The benefit of this reflects on issues of power consumption, as well as issues of bandwidth overhead, particularly when there is little or no mobility.

However, as hops are added to the packet header field, there is an increase of the packet overhead during route discovery. This overhead grows in proportion to the number of nodes that a RREQ has to transverse through before reaching its destination node. As a result, the advantages seen above when little network traffic or mobility is taking place come in a package of disadvantages involving higher overheads upon wanting to exchange information.

Finally, much of the protocol's ability to learn and adapt to a new communication environment, presents also a problem in terms of security. As an example, a MN acting in promiscuous mode will have the ability to extract a lot of information regarding the network and the nodes that constitute it, prior to launching an attack. This is further amplified by the fact that in DSR there is only a single error packet type that is sent regardless of the type of failure occurring in the link between the nodes in question.

2.3 Destination Sequenced Distance Vector (DSDV)

The flip side of on demand routing protocols groups table driven designs. This category of protocols operates on the assumption that every node should have an up to date view of the entire network at any one time. In order to achieve this, table driven routing relies heavily on the transmission of updates to any noticeable change in topology. Key elements in their classification involve the different routing tables that each node must store, as well as methods of transmitting updates throughout the network.

2.3.1 Description and origin

The Destination Sequenced Distance Vector (DSDV) routing protocol [36, 28] is a hop-by-hop distance vector design [37] relying on the table driven approach. It bases its operation on the Fulkerson method [38], also known as the Bellman-Ford algorithm [39]. DSDV improves on traditional distance vector, by guaranteeing loop freedom in suggested table routes.

Each MN is required to maintain a routing table storing all of the possible destinations within the network, as well as the number of hops to each one. In order to guarantee loop free routes, each entry is tagged with a sequence number. This number shows the freshness of the route; the higher the sequence number, the more favourable the route is. In deciding on which route to select, if two routes carry the same sequence number, the route with the least number of hops is preferred. Finally, when a node detects that an advertised route within its own routing table is no longer valid, it increases the sequence number for that route, advertising it to other nodes as a route with an infinite hop count.

Routing table updates are transmitted in regular time intervals throughout the network. As a control mechanism towards the potentially vast amounts of network traffic that can be generated from this process, two types of packets exist. The first, known as a full dump [36], carries all available routing information and is generally transmitted during periods of partial to little movement. For time periods of high mobility, smaller packets in size are used, aiming to relay only information that has changed since the last full dump in an incremental update. The classification of these two types of updates takes place

with respect to Network Protocol Data Units (NPDUs). For the incremental updates, data transmitted should fit into a standard size NPDU, while for full dumps multiple NPDUs might be required.

Every broadcast of a new route contains the address of the destination, the number of hops required to reach it, the sequence number that the broadcasting node has labelled that route with and a new sequence number unique to the re-advertising of that route. Each MN is also required to delay advertising a new route, thus allowing for a certain amount of time labelled as settling time [36], so that the route with the best metric can be discovered and circulated.

2.3.2 Protocol characteristics

DSDV mirrors the distance vector technique as that has been applied to wired topologies, adding a number of adjustments to better suit ad hoc networks. These include updates circulated to the entire network upon any noticeable change in topology, as well two types of available message types for broadcasting such updates.

The dependency of this protocol on incremental updates yields a necessary time interval of convergence before a route can be used. Even though this convergence time can be considered negligible in a static network, it represents one of the biggest drawbacks of DSDV [40] in wireless environments. Due to the frequently changing network topology, this time of convergence will have as a direct result an increased number of dropped packets prior to the full detection of a new route.

Finally, even though the overhead is minimised with the two types of route table dumps that can take place by every node, the requirement that every change in topology is broadcasted to the network in its entirety is still present. As a result, the overhead will at all times remain high, relative to other protocols that require less or more partial information to be transmitted in a similar scenario.

2.4 Ad hoc On demand Distance Vector (AODV)

Building on DSDV, the Ad hoc On demand Distance Vector (AODV) routing protocol [41, 42, 43, 44] combines the characteristics seen in the previous section with a source initiated on demand design. This protocol is reactive, only requesting a route when one is needed, as opposed to proactive implementations which attempt to keep a current view of the entire network at all times.

2.4.1 Route tables, route discovery and route maintenance

Every MN utilising AODV is required to keep a route table. Each entry in this table has to contain each of the fields described in table 2.1 [41, 44]:

Table 2.1: The fields that each entry within the routing table of AODV must carry.

Field	Description
Destination IP address	The unique address of the destination node
Destination sequence number	Sequence number for this destination, ensuring that the route is loop free
Hop count	The number of hops to the destination
Next hop	The neighbour that has been selected for forwarding packets to the destination MN of this entry
Lifetime	The time for which the route is considered valid
Active neighbour list	Neighbour nodes that are actively using this entry
Request buffer	Buffer making sure that each entry is processed only once

When a MN, say S, desires to send a message to another node, say D, and provided that it does not have a valid route to that destination, it initiates a route discovery process. This is achieved through the transmission of a RREQ packet broadcast to all its neighbours, who are in turn obliged to forward this request until either D is located, or a node with a “fresh” route to D is found. This is illustrated in figure 2.2a. Unlike DSR, the RREQ packet maintains a constant size with its header not carrying incremental MN IDs of the nodes visited.

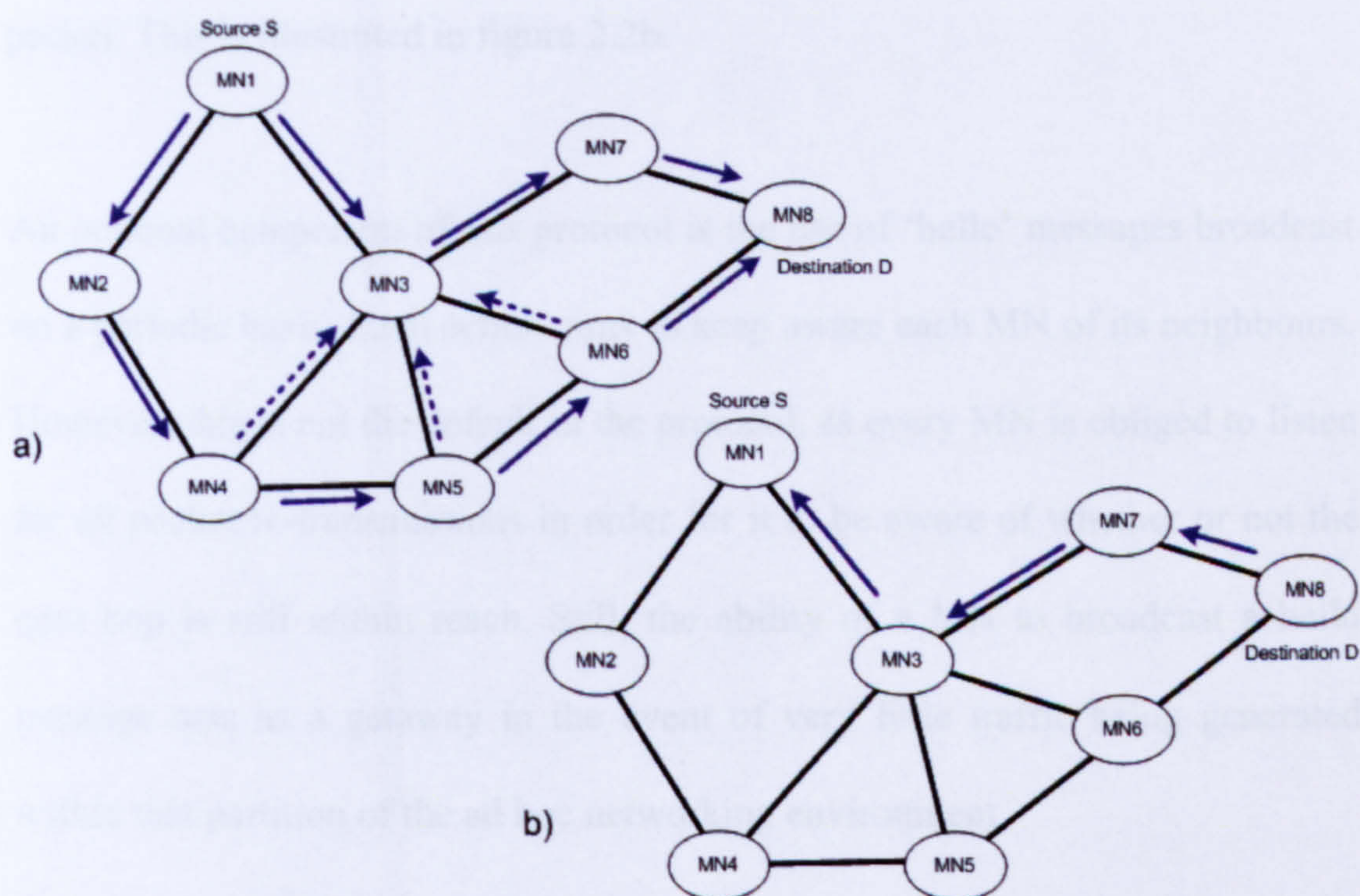


Figure 2.2: The a) transmission of an AODV RREQ packet from source S, targeting destination D; b) propagation of the route reply from D back to S.

Instead, during the route discovery process, each intermediate node is required to record an entry in their routing table of the first node that they received the packet from. This has a dual purpose; it establishes a reverse route for the packet to follow during the reply phase and also allows for the MN to disregard any later received copies of the same packet. Keeping a record of such information guards intermediate nodes from processing the same request more than once.

The requirement in AODV for a node to offer a route reply goes beyond having an available route to D. Not only must a MN know of a way of reaching D, but also the proposed route must have a destination sequence number greater than or equal to that contained in the RREQ packet. If this condition is not met, a route cannot be classified as “fresh” and the MN cannot advertise this route. Provided that a fresh route is found, it is broadcast back to S, following the entries that each neighbouring node has stored with respect to the origin of the request packet. This is illustrated in figure 2.2b.

An optional component of this protocol is the use of ‘hello’ messages broadcast on a periodic basis. Such action aims to keep aware each MN of its neighbours. However, this is not the default in the protocol, as every MN is obliged to listen for all packet re-transmissions in order for it to be aware of whether or not the next hop is still within reach. Still, the ability of a MN to broadcast a hello message acts as a getaway in the event of very little traffic being generated within that partition of the ad hoc networking environment.

Finally, when a node detects that a specific route is no longer valid, it broadcasts a link failure notification message [41]. This message is simply a RREQ packet with an infinite metric towards the node which broke the path within the known route. Every node must propagate each failure notification to the nodes that are actively using this route. This is the reason behind AODV using an active neighbour list (seen in table 2.1) enabling it to keep track of neighbouring nodes utilising specific routes.

2.4.2 Protocol characteristics

One of the main characteristics of this protocol is the trade-off that it utilises for further reducing the network overhead. To minimise the number of routing messages sent across the network, AODV requires MNs to store information about generated neighbouring traffic in their routing tables. Furthermore, it takes into account the history of proposed routes by utilising sequence numbers. For these two reasons, it offers a greatly reduced number of routing messages in the network.

Even though the sequence numbers that AODV uses prevent loops from existing within proposed routes, their use implies a strong element of synchronisation, which can give rise to other problems. As an example, consider a network consisting of a number of partitions, which remain isolated from one another for long periods of time. In the event of the two partitions merging, any route advertised will carry a sequence number that will be either much too old or far too new to some of the MNs. This yields that the use of sequence numbers can give rise to other problems, outside those that they have been implemented to solve.

Finally, any triggered route replies involving failure notification messages have to always make their way from the MN of failure to the sender. Even though this task is further simplified by transmitting this information only to the affected senders, it still implies that in a highly dynamic topology the network overhead for the protocol would increase. As AODV triggers one RREQ packet for every node in each node's active neighbour list, it is more than likely that for

every failure more than one notification will arrive at any intermediate node. A solution to this problem would involve a more collective approach where neighbouring nodes could group such notification messages, thus controlling the number that gets transmitted from a particular partition within the network.

2.5 Temporally Ordered Routing Algorithm (TORA)

The Temporally Ordered Routing Algorithm (TORA) [45, 46] is a distributed routing protocol that is highly adaptive and favoured for greatly mobile ad hoc networking environments. It falls in the category of source initiated on demand protocols and bases its operation on the concept of link reversal [47]. The main feature behind its design lies in the fact that nodes are only required to keep up to date routing information about nodes that are 1-hop away. The properties of the protocol describe three basic states: route creation, route maintenance and route erasure. We shall examine each of these in the section below.

2.5.1 Route creation, route maintenance and route erasure

TORA requires every node to carry a “height” metric for both the stages of route creation and maintenance. In order to direct traffic to the labelled destination MN, the protocol establishes a Directed Acyclic Graph (DAG). The rule for propagating messages applies to the DAG formed: all messages in the network flow downstream, from a node of a higher height, to a node of a lower height. This is illustrated in figure 2.3. This particular protocol is referred to as an algorithm because of the fact that it builds on Internet MANET Encapsulation Protocol (IMEP [48]). TORA focuses only on the underlying routing mechanism, relying on IMEP for its remaining functionality. For this reason, the protocol is often referred to as TORA/IMEP [49].

In the process of route discovery, when a node with no downstream links requires a route to a destination, it broadcasts a query (QRY) packet. This packet propagates through the network, following MNs of less height until it reaches a MN that has a route to the destination or the destination itself. As a next step, this node is required to broadcast an update (UPD) packet containing its height value. Every node receiving this packet has to adjust its own height to a value higher than the one advertised in the UPD packet. In true pyramid fashion, each node upon receiving a UPD packet has to recursively transmit a UPD packet with its height value. As this process continues, it creates a path between the source and destination, often yielding multiple routes between the two.

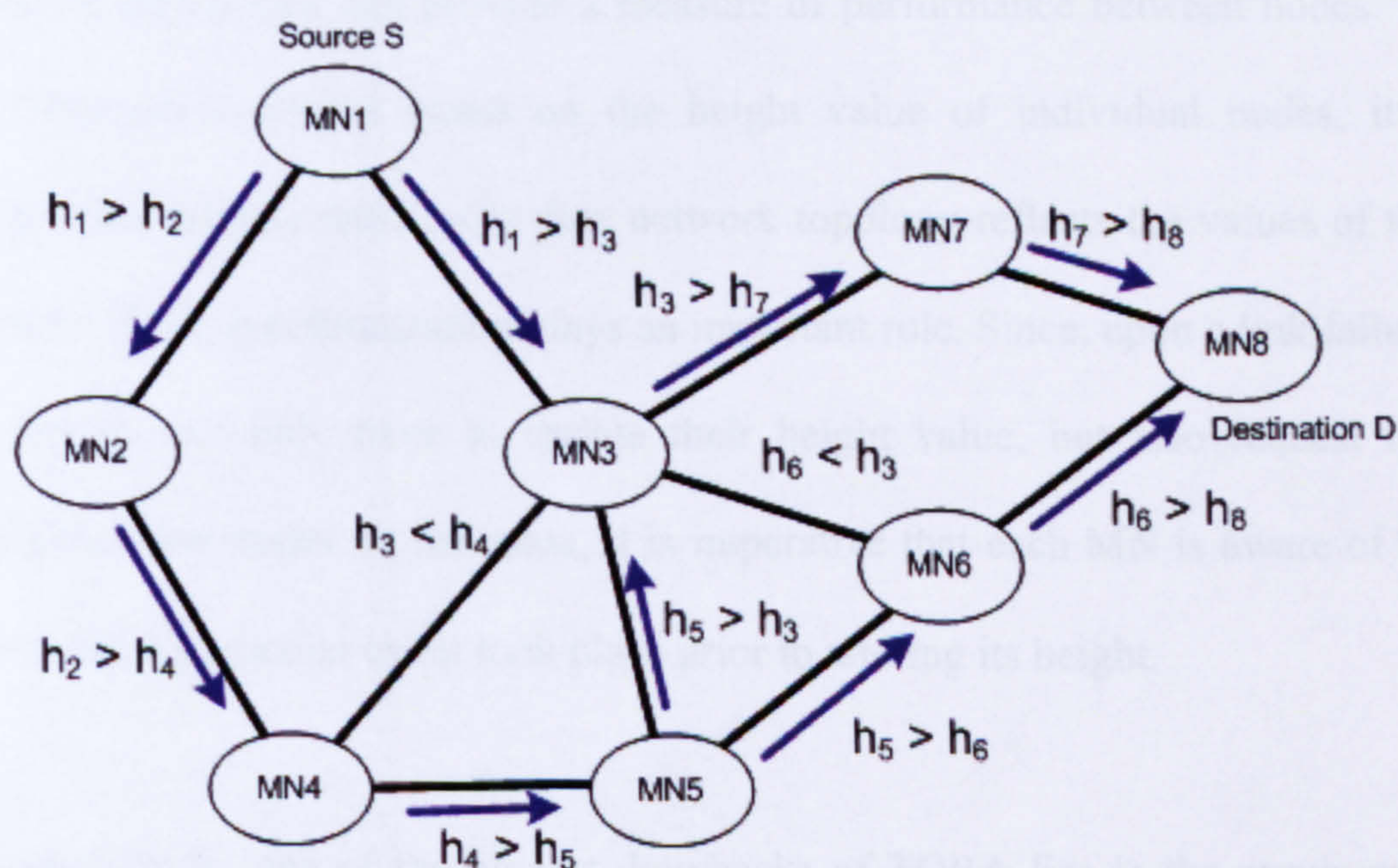


Figure 2.3: The propagation of a route request in TORA from source to destination, based on the respective “height” of each MN. The resulting graph represents a DAG.

In times of high mobility, the resulting DAG route established through the above described process is often broken requiring further update. Upon failure to transmit, the last downstream MN generates a new height reference that is established in a similar way as that taking place during route discovery.

Finally, another unique element of TORA lies in the ability of each MN to request from a neighbouring node to erase a specific route from its route cache. This feature is implemented through the use of CLR messages, carrying information about which routes are considered no longer valid. As an example, upon the detection of a network partition, all routes involving nodes outside the fragmented subnet are cleared from the route cache of “inside” MNs. This takes place through the propagation of CLR packets among the subnet, from attempts to communicate with nodes outside it.

2.5.2 Protocol characteristics

The key characteristic in the implementation of TORA lies in the utilisation of a relative metric that can provide a measure of performance between nodes. As information is routed based on the height value of individual nodes, it is important that the most up to date network topology reflects the values of this metric. Thus, synchronisation plays an important role. Since, upon a link failure, nodes do not only have to update their height value, but also request that neighbouring nodes do the same, it is imperative that each MN is aware of the time that a particular event took place prior to altering its height.

Consequently, one of the biggest drawbacks of TORA lies in the requirement for clock synchronisation between communicating nodes. This is seen as a necessary condition for the algorithm to function correctly, which lies outside the protocol specification and can simply be taken for granted a priori. Even though existing standards such as the Global Positioning System (GPS) can satisfy this prerequisite, it forms a dependency for routing that supersedes a stand alone protocol implementation.

As every MN receives routing updates from all its neighbouring nodes and without prioritising incoming packets, fast alterations in the height value of each MN can take place. The occurrence of such an effect will be mostly noticed during concurrent detections of network partitions from multiple nodes. This will in turn trigger each MN to attempt to fix the problem by advertising route erasures as well as attempting to rebuild specific links to other destinations. As a result, upon a significant change in network topology, height values can vary by a significant amount, before stabilising to a set value.

2.6 Zone Routing Protocol (ZRP)

Moving away from protocol designs that focus on utilising between the choices of characteristics described in section 2.1, the Zone Routing Protocol (ZRP) gives a hybrid approach to routing. The specification of this protocol [51, 52, 53] combines the reactive approach of querying for a route on demand, with the proactive slant of attempting to keep an up to date view of network topology. To achieve this, ZRP partitions the network in a number of routing zones and deploys two separate sub-protocols that are used between, as well as inside the zones specified.

2.6.1 The concept of routing zones

A routing zone is defined [51] as a set of neighbouring nodes that are, at maximum, a certain number of hops away. This maximum distance, measured in hops, is referred to as the zone radius. To allow for communications to take place, ZRP specifies the Intrazone Routing Protocol (IARP), which operates within a given zone and the Interzone Routing Protocol (IERP) used for finding routes between different zones.

For MNs within the same zone, IARP is used. This sub-protocol is not defined; it simply assumes a proactive architecture, such as DSDV or a similar table driven implementation. The specification of ZRP does not require all zones to operate using the same IARP. The sole requirement is that upon a change in network topology all nodes within the affected zone are notified. This can result in nodes changing zones and thus zone protocols depending on their mobility.

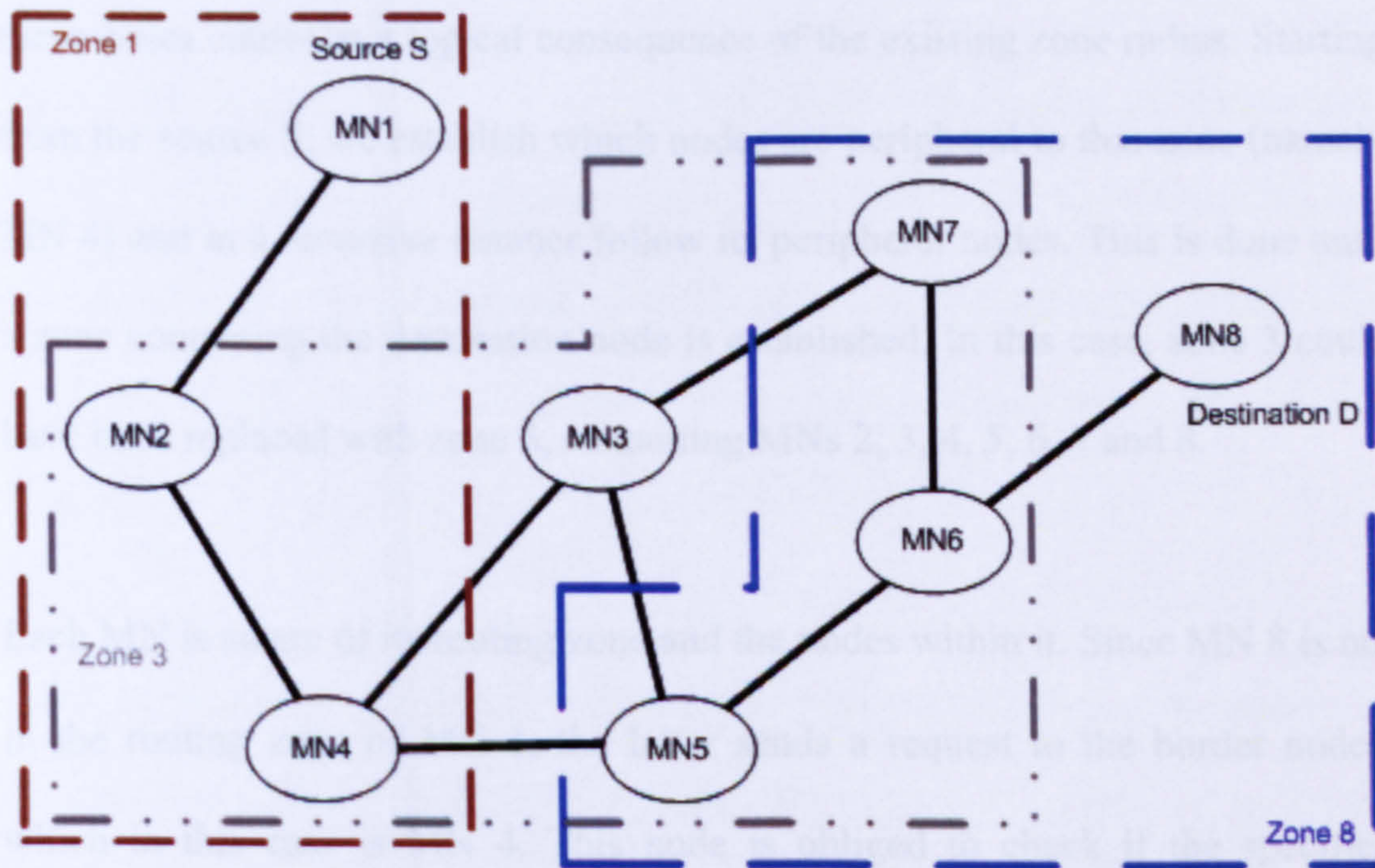


Figure 2.4: The functionality of ZRP, separating the network into respective zones for source node S and destination node D.

Provided that the destination MN does not lie within the same zone as the source, IERP is deployed. This protocol bases its operation on RREQ packets which get broadcast throughout the different zones, recursively, until the requested route is found and a reply is sent back to the source. For this process IERP employs the Bordercast Resolution Protocol (BRP), included within the ZRP specification. BRP bases its operation on transmitting any RREQ packets to the border MNs within a given zone and attempts to keep track of which nodes do act as communicating borders within existing zones.

As an example of the functionality of ZRP, consider the networking environment with the respective links illustrated in figure 2.4. Source node, say S (MN1) wants to establish a route with destination node say D (MN8). Assuming a zone radius of 2, since both nodes do not lie within that radius, we can see that three separate zones surface, zone 1 with MNs 1, 2 and 4; zone 3 with MNs 3, 2, 4, 7, 6 and 5; zone 8 with MNs 8, 7, 6 and 5. The derivation of these zones comes as a logical consequence of the existing zone radius. Starting from the source S, we establish which nodes are peripheral to that zone (namely MN 4) and in a recursive manner follow its peripheral nodes. This is done until a zone containing the destination node is established. In this case, zone 3 could have been replaced with zone 5, containing MNs 2, 3, 4, 5, 6, 7 and 8.

Each MN is aware of its routing zone and the nodes within it. Since MN 8 is not in the routing zone of MN 1, the latter sends a request to the border nodes, which in this case is MN 4. This node is obliged to check if the specified destination is within its routing zone. In the event that it is, a route reply is generated. As in our example this is not the case, the request is further forwarded to the border nodes of MN 4, which are MN 7 and MN 6. As the destination is within the routing zone of both these zones a request reply packet is generated and sent back to the source following the same principle.

All requests made within a single zone utilise the IARP. For requests that cross zones, such as a route request from MN 1 to MN 8, IERP is used. Finally, for both a route request and a route reply, to prevent requests from entering previously queried zones, a process request stack is used. Each node keeps track

of previously queried packets, performing a comparison for all incoming traffic. If a node receives a request that it has previously processed, the incoming packet is simply dropped.

2.6.2 Protocol characteristics

The hybrid design of ZRP bases its operation on establishing a local and global network view. On the local level, every MN belongs to a zone of which it knows all the members; on the global level, the aspect of exchanging information between two distant nodes is divided into the task of routing packets between different zones. What enables us to distinguish between the two views is the zone radius defining the number of hops that constitute each zone.

From the protocol specification, there is no mention regarding the decision of the size of the zone radius; it is suggested [51] that this should be selected by a central administration, depending on the requirements of the MANET. Since the usage of this protocol is targeted at a constantly changing topology, the optimum value for the zone radius can vary. However, adjusting its value is something that cannot be done dynamically and therefore holds a potential drawback for ZRP.

Another issue lies in the decision not to include the full specification for intra-zone communications. As different zones can utilise different proactive protocols, any rapid change in the network topology can result in a MN attempting to communicate to another node using constantly changing rules. Apart from the obvious issue of compatibility (all MNs are required to support

any different protocol that they might encounter on the network) this can increase the network overhead by a significant amount. Furthermore, as a change in topology can have an effect on more than one zone, issues of time latency can surface. As nodes adjust to the settings of their new zone, a time window will be required before they will be fully aware of the rules utilised in their new neighbourhood.

2.7 Cluster Gateway Switch Routing (CGSR)

The Cluster Gateway Switch Routing (CGSR) protocol presents an alternative implementation to the flat structures that we have so far seen. This protocol [53, 54], utilises a hierarchical structure dividing the MNs present in a number of overlapping or disjoint clusters. Each cluster has a cluster head, responsible for maintaining information about its members. In order to establish a route within a cluster, the head MN is consulted in a reactive manner, establishing a route on demand.

2.7.1 The interoperability of clusters

In order for this protocol to function correctly, a cluster head algorithm is used. This serves a dual purpose among neighbouring nodes; firstly, it informs them which cluster they belong to and secondly, which node is acting as a local authoritarian MN. The algorithm functions on the principle of electing the node with the smallest ID as a cluster head. By transmitting hello messages, which do not get forwarded beyond a single hop, each MN computes the node acting as a local cluster head. A node is classified as a member of a cluster, if it has a bi-directional link with the cluster head. To allow for communications to spawn

the entire network, each node can be a member of more than one cluster. This, depending on the physical topology, prevents partitions from being formed. Every node that belongs to two or more clusters acts as a gateway between them. Each cluster head is responsible for receiving all incoming traffic for the cluster, which then gets forwarded respectively, either to nodes that are part of the cluster group, or to nodes acting as gateways. A vital condition for this functionality is the notification from a MN to a cluster head of its ability to communicate with another head. An example of this operation is illustrated in figure 2.5.

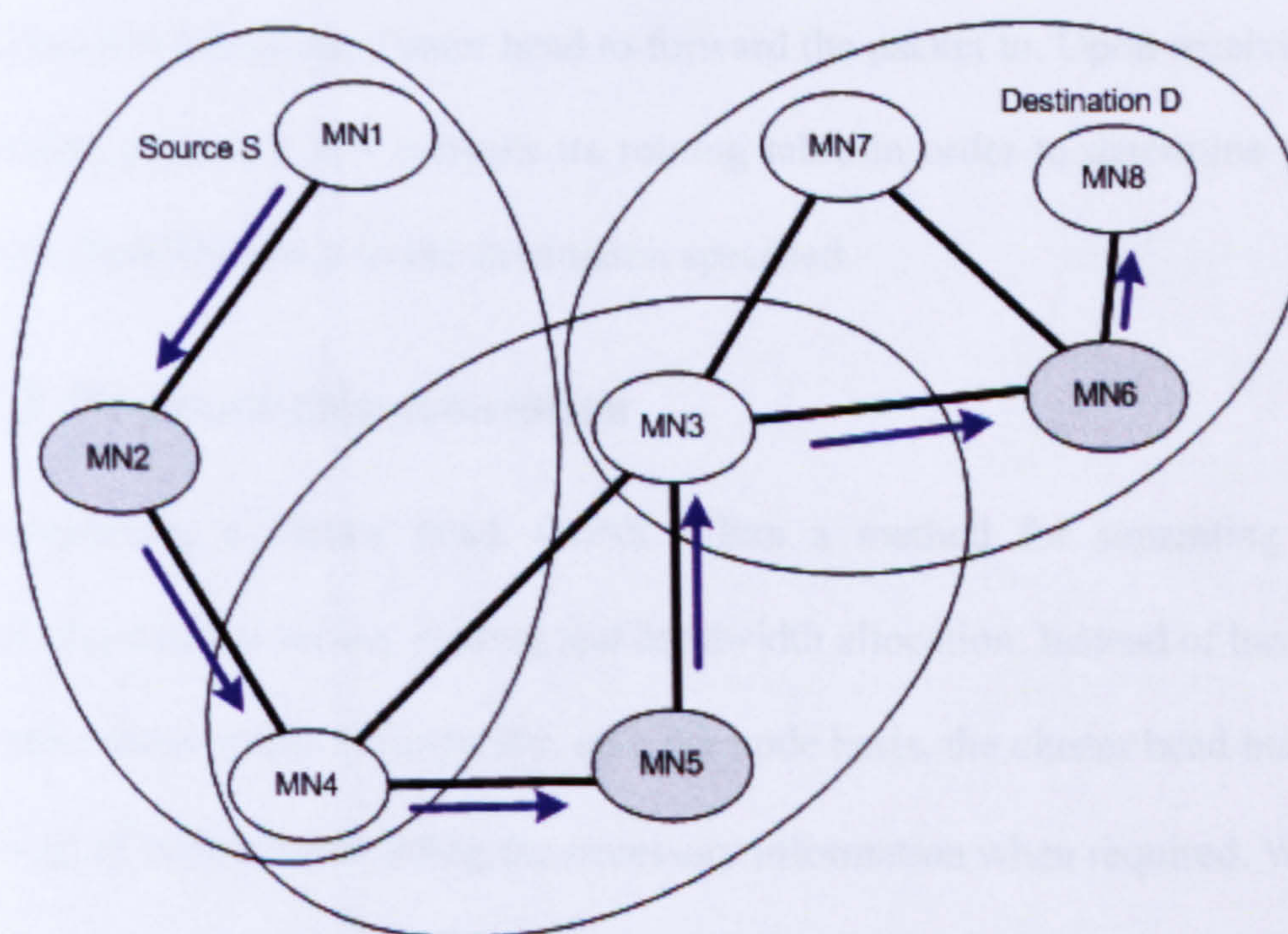


Figure 2.5: The process of routing packets from the source MN 1, to the destination MN 8 via cluster heads, utilising CGSR.

In this example, the source node S, wants to transmit to the destination D. For this, it first consults its cluster head MN 2, which in turn forwards the request to MN 4 that acts as a sole gateway between the two clusters. From MN 2, the packet gets forwarded to the respective cluster head (MN 5), which in turn forwards it to the other gateways, excluding the incoming node. In this manner

the packet reaches the destination. A reply from node D back to S travels through the network in a similar fashion to that described above.

The authors of CGSR [51] state that their protocol utilises a version of DSDV, modified to best fit a hierarchical network structure. Each node is required to store a cluster member table, which similarly to the routing table of DSDV is broadcast periodically across the cluster. In addition to the cluster table, each node is required to store a routing table used for determining the next hop for reaching the destination. The purpose of this lies in the event of a node having to select more than one cluster head to forward the packet to. Upon receiving an incoming packet, a MN consults its routing table in order to determine which cluster head lies closer to the destination specified.

2.7.2 Protocol characteristics

By deploying a cluster head, CGSR offers a method for separating tasks involving channel access, routing and bandwidth allocation. Instead of having to consider these issues individually, on a per node basis, the cluster head manages a group of nodes, transmitting the necessary information when required. When a node wants to communicate with another node, it consults its local authority MN and transmits packets based on the feedback it obtains. This process establishes a trade-off; on the one hand it offers a number of advantages with regards to minimising network overheads and on the other, it leaves the remaining cluster nodes unable to communicate in the event of not being able to access the cluster head.

Furthermore, when utilising a cluster head scheme, any frequent change in topology, resulting in the partitioning of the cluster will require a cluster head re-allocation to be performed. Depending on how often this will be required and how many clusters it effects, it can result in devastating results with respect to performance; MNs will be busy establishing the infrastructure that enables them to exchange information, instead of actually using this to process incoming information.

For this reason CGSR limits the ways in which a cluster head can be established. By utilising the Least Cluster Change (LCC) clustering algorithm, it allows for head reselection only in the event of two cluster heads establishing a bi-directional link, or a node moving out of the range of its cluster head. LCC creates a scenario that limits the way in which a node can become a cluster head; for this the algorithm takes from the dynamic allocation of one cluster head, when and as necessary. On the other hand, these two conditions are necessary if we are to guarantee the correct functionality of this protocol.

2.8 AntNet

Having reviewed a number of protocols that base their operation on utilising the different characteristics presented in the first section of this chapter, AntNet proposes an agent-based adaptive approach to routing. Spawning from the field of emergence, this algorithm [55 – 59, 66] takes the survival instincts observed in ant colonies and applies them to the problems of wireless routing.

The framework of operation within the protocol that AntNet encapsulates, uses a number of characteristics that lie outside traditional routing techniques. In this

section, we aim to provide an overview of the operation of AntNet, without questioning the theoretical framework behind it. Chapter 3 (and in particular section 3.1) aims to introduce the overlapping fields and theoretical components behind this implementation.

2.8.1 Understanding ant colonies

Ants in their struggle for survival are characterised by their reproductive division of labour, their cooperative brood care, as well as the presence of a sole worker caste. It is these three characteristics [60] that classify them along other social Hymenoptera (ants, bees and wasps) as highly organised insects that in a single word can be described as *eusocial*.

The social organisation of these insects is reflected in the complex form in which their colony is managed. Despite lacking a central administrative authority for guiding the allocation of tasks and resources [15], ants appear to have the ability of overcoming a number of difficulties in their quest for food. As simple as this chore might appear, what we must not forget is that these insects do not have the ability to communicate with each other directly. As a result, the question that rises relates to how they actually manage to cooperate and achieve such a complex and organised behaviour for the whole colony.

The key in their interaction is the well-studied phenomenon of foraging and recruiting [61 – 65] embedding the concept of Ant Colony Optimisation (ACO). This method of communication explains how ants explore their surrounding environment scavenging for food, managing to, not only find their way back to their nest, but also indicate the food source to neighbouring insects. Each ant

has the ability to release a noticeable chemical substance at their current location called pheromone. This acts as a crowd-puller to other ants, which in turn, provided they spot something of interest in the vicinity will release more of this substance to generate further attention. To our understanding today, a single ant does not have any control over the amount of pheromone that it releases in the environment, as this is done naturally and depends on the excitement and stimulation that is received from the environment. Consequently, ants end up moving within trails of pheromone near their nest. The closer to the nest an ant is the stronger these trails are due to the presence of more ants. As pheromone evaporates in time, each trail by default has a set time of existence prior to either disappearing or being remapped by the presence of more ants running across it. The resulting behaviour that ants possess, by having the ability to release and also feel attracted to pheromone, describes the phenomenon of foraging and recruiting.

As an example of this, consider an ant encountering an obstacle in their way towards a pre-defined (in terms of an established trail) food source. Initially, this ant will randomly pick a new direction of travel. Depending on the fractions of pheromone that it deposits, it will effect the decision of other ants trailing along and stumbling on the same obstacle. If the leading ant manages to randomly find a new path to the food source, it will release more pheromone along its trail that will in turn attract other ants. If this does not take place, then the second ant will pick up a new vanishing trail, having to in turn select a new random direction of movement. This process will carry through in a recursive manner, until the food

source is reached, or interest for that food source is lost because of all the trails having vanished.

Through this process, a natural system comprised of entities deprived of the ability of direct interaction and communication, manages to have an information flow serving the purposes of survival for an entire ant colony.

2.8.2 Ant based routing

As distant as the process that ants use to survive might seem to mobile ad hoc routing, there are a number of protocol designs that use the pheromone approach of foraging and recruiting. The common denominator that allows researchers to link the two comes from the fact that in both cases a system or network of simple interacting entities needs to robustly solve the problem at hand with as little direct communication as possible. It is this fact that connects two fields that would otherwise appear as quite far apart.

Building on the metaphor that ants looking for food can also represent packets looking for destinations, the question that arises is to what extent can routing be abstracted as an emergent behaviour from the interaction of packets travelling through the network. To help us answer this question, we proceed by reviewing one the pioneering protocols of this type: AntNet.

In the process of prescribing incoming packets to an outgoing path in an ad hoc networking environment, AntNet applies the concept of optimisation within the ant colony (ACO) [57, 61] to wireless routing. This algorithm, as it was first presented in [66], introduces control packets on the network, named ant packets.

Data is forwarded on the network based on next-hop probabilities and with each MN looking for the minimum cost path joining the source and the destination. Ant packets are responsible for discovering and maintaining routes, using node to node trip times to adjust next hop probabilities. Through this mechanism, ant packets follow ant behaviour in colonies, “selecting” their direction of movement based on a higher probability ratio. It is this probability value that represents the pheromone within the MANET.

Each packet travels through intermediate nodes on the network, being pointed towards its next hop based on a greedy stochastic policy. Within the packet, a list of visited nodes is maintained, as well as the time that has elapsed to arrive there. Ant packets launched towards the destination are referred to as forward packets. Each time a forward packet arrives at an intermediate node, the stochastic process, based on uniformly selecting the next hop with a small exploration probability, is launched. If the node selected as a next hop has already been visited, the next best hop available is chosen.

When a forward ant packet arrives at its destination, a backward ant packet is launched towards the source. This packet follows the reverse path to that found in the arriving source packet. At each MN, this backward packet updates the routing of each node it passes through. This update involves next-hop probabilities to the nodes that the packet has already visited, as well as trip times (in terms of the mean and the variance) to the destination. Forward packets carry the same priority as data packets on the network experiencing the

same congestion and delay as data, while backward packets carry a higher priority. Once the packets arrive back at the source, they are simply dropped.

Through this process, a dynamic alteration in the probabilities that each node carries in its routing table is offered, representing the continuous change in pheromone trails within an ant colony. Based on this approach, ant packets are sent between source and destination pairs, testing the connectivity along that path. This is done asynchronously, by selecting destinations at random, in an attempt for each node to have a more up to date view of the routing conditions on the network. Due to the behavioural characteristics of ant packets and, despite the fact that they simply correlate control information between nodes, these packets are often referred to as mobile agents or ants in references to AntNet.

2.8.3 Similar protocol designs

Expanding on ant based routing, a number of other protocols exist that utilise the concept of foraging and recruiting in a similar way. Starting from circuit-switched telephone networks, Ant Based Control (ABC), first introduced in 1996 [67] aims to optimise the routing of calls placed between different telephone exchanges. Despite being applied to wired networks, this protocol bases its operation on ant packets leaving a pheromone trail, being launched at regular time intervals towards randomly selected destinations. When a call is placed, a circuit must be established between the caller and the destination. The highest probability hop is followed from the source outwards; if a circuit cannot be established, the call is blocked. In this fashion, when routes are sufficiently short, actual calls are placed onto the network.

Another design, this time for MANETs, is Ant-Colony based Routing (ARA). This algorithm [68] employs a flooding scheme to find destinations, using pheromone trails to determine next hop probabilities. Similarly to AntNet, forward and backward ants are used for discovering routes between the source and destination. For each forward ant that is flooded through the network and received by an intermediate node, a backward ant is returned. As a result, reverse routes are established automatically as forward ants move from one node to the next towards the destination. Upon the occurrence of a route failure an automatic retransmission of the packet takes place, backtracking to the previous node if necessary. Despite this scheme not being loop free, it offers a way of checking for loops through the individual pheromone probabilities that each node has.

A final design, also for MANET routing is Termite [69]. This algorithm, further builds on the pheromone characteristics of AntNet, attempting to minimise the control traffic generated on the network. Similarly to other routing protocols presented in this section, Termite also uses the pheromone approach to in turn offer next hop probabilities for data packets. Each packet is forwarded probabilistically, based on the amount of destination pheromone found on each neighbour link. The unique aspect of this algorithm lies in a neighbouring node being dropped from a node's pheromone table if transmission fails. Consequently, if a node does contain a needed destination node in its pheromone table, a route request is issued via a RREQ packet. It is this process,

very much resembling the route request of DSR and AODV, which allows for further reduction of the control packet overhead on the network.

2.9 Methods of comparison

From the protocols presented in the previous sections, a number of different approaches to routing have been presented. Starting from table driven designs, to on demand and adaptive agent based routing, each protocol builds on the characteristics presented in section 2.1, adding further to these through its own unique design. In this section, we aim to provide a number of comparison characteristics for each of the protocols presented. As these protocols have been analysed theoretically, the characteristics that follow are derived from theoretical and qualitative analyses based on what properties the protocols have.

2.9.1 Protocol complexity

As an initial comparison for measuring the performance, we focus on the complexity of the communication process. This is further divided into two categories; time complexity, defined as the number of *steps required* to perform a protocol operation, and communication complexity, defined as the number of *messages needed* to perform a protocol operation [70, 71]. For each of the two, we consider the complexity during protocol initialisation, as well as protocol failure. This distinction, also seen in [28], is useful when comparing the protocol's response to one or more topological changes. Commonly, it is source initiated on-demand routing designs that mostly alter their behaviour after the occurrence of a link failure. Consequently, this measure of performance not only takes into account how each protocol operates in terms of the packets

transmitted and steps required, but also how this behaviour changes upon one or more link state changes within the MANET.

The measure of comparison is performed with respect to the order O of the most significant characteristic affecting the performance of the protocol. This is a direct derivative of the way in which algorithm complexity is measured [72]. The complexity of a protocol is measured by expressing either the number of steps (for time complexity) or the number of messages (for communication complexity) as a function of one or more of the affecting characteristics. Commonly the values in any of these metrics aim to represent the worst case behaviour. As an example, a protocol of complexity $O(2d)$ is of order, twice the diameter d of the network, as seen for DSR, AODV and TORA in Table 2.2.

From the protocols that we have reviewed in this chapter, the characteristics that affect their complexity are: the number of MNs N on the network, the diameter d of the network, the number of nodes x affected by a topological change, as well as (in the case of hierarchical protocols) the height of the routing tree h that a cluster forms and the average number of nodes M each cluster has. Naturally, more complex designs can be dependant on more versatile characteristics. For the case of ZRP, we also use the average number B of border (gateway) nodes within a cluster. From the existing characteristics, it is important to have the ability to distinguish between them. As an example, the difference between the network diameter and the number of nodes lies in the way the nodes are spread across the network; the diameter represents the maximum number of nodes in the longest path. Each one of these characteristics represents a unique measure

of performance, which, depending on the design, could carry more importance with respect to others.

Table 2.2 shows a comparison of the time and communication complexity of the different protocols presented in this chapter. For DSR, TORA, AODV, DSDV, ZRP and CGSR the original tables as well as further explanation for each protocol can be found in [28, 30]. This table lacks the time and communication complexity of AntNet. As this protocol bases its operation on simulating pheromone trails, the parameters that affect the way in which ants move across the network lie outside the conventional characteristics presented in this table. A study on issues of complexity for AntNet and other ant based algorithms can be found in [73], with further mathematical background presented in [74].

Table 2.2: Comparison of the complexity characteristics of the protocols reviewed.

Time Parameters	DSR	DSDV	AODV	TORA	ZRP	GCSR
Time complexity (initialisation)	$O(2d)$	$O(d)$	$O(2d)$	$O(2d)$	$O(M) / O(2d)^*$	$O(d)$
Time complexity (failure)	$O(2d)$ or 0 ^{**}	$O(d)$	$O(2d)$	$O(2d)$	$O(M) / O(2d)^*$	$O(d)$
Communication complexity (initialisation)	$O(2N)$	$O(x=N)$	$O(2N)$	$O(2N)$	$O(M)^* / O(2Bd)$	$O(x=N)$
Communication complexity (failure)	$O(2N)$	$O(x=N)$	$O(2N)$	$O(2x)$	$O(M)^* / O(2Bd)$	$O(x=N)$
<p style="text-align: right;"> d : Network diameter h : Height of the routing tree N : Number of MNs on the network M : Average number of nodes within each cluster x : Number of nodes affected by a topological change B : The average number of gateway nodes within a cluster </p> <p>* Intrazone / Interzone ** In the case of a cache hit</p>						

From this table, we can see that the complexity of a protocol can be dependant on different characteristics, based on the way that it has been designed. Moreover, the order of either the communication or time complexity does provide an impartial method of merit, regardless of the simulation scenario at hand. For more complex protocol designs, the characteristics used also become more complicated. Even though this method of comparison does not offer a direct measure of performance between different designs, it does yield the dependencies that each protocol uses.

2.9.2 Single based characteristics

The opening section of this chapter presents a number of characteristics that enable us to classify different routing protocols. Each of these can further produce a way of not only distinguishing but also of comparing them. This process, when attempting to use the method of comparison as a measure of performance, can lead into a number of pitfalls. This section aims to provide the necessary clarification between a design characteristic and a comparison mechanism in wireless routing protocols.

As an example we refer to the characteristic of a flat versus a hierarchical structure. From the protocols presented, CGSR has a hierarchical structure establishing clusters within the MANET, while AODV, for example, has a flat structure. Similarly, in both [28] and [30] that we have referred to from the previous section, one of the key characteristics when comparing the different protocols is their routing philosophy: this can be either flat or hierarchical. In this instance, the method of comparison signifies the design criteria.

Such characteristics that are identical in both the design as well as the comparison stages, we label as single based and continue to question with caution. As the information used for the design of the protocol is directly passed for its comparison with respect to other designs, no significant conclusion can be derived regarding the evaluation of its performance. This criterion simply reminds the reader of the design method that was used.

Ergo, for each characteristic presented in the classification of a routing protocol that helps us to further quantify each design, a single based characteristic that can be used for comparison exists. This technique should be avoided, as it does not give rise to an independent way of assessing the success or failure of the resulting design.

2.9.3 Typical characteristics

For the comparison of routing protocols, further to protocol complexity and single based characteristics, a number of quantitative metrics can be used to measure the performance of any routing protocol. These measures, despite being dependant on the simulation scenario at hand, provide the most effective way of distinguishing between different protocols and accessing their performance in situations that are as realistic as possible.

Following the ongoing efforts of the Internet Engineering Task Force (IETF) [75] working group on MANETs [76], a number of standards and specifications relating to ad hoc networking technology and simulation have been peer-reviewed for

public review. Corson and Macker put forward RFC⁴ 2501, in which they propose the following quantitative metrics for assessing the performance of any routing protocol [77]. These are:

- **End to end data throughput and delay**

Statistical measures of the amount of data that can be transmitted between nodes, as well as considerations for any delay between transmissions. These measures of effective routing are performed with other protocol designs as benchmarks.

- **Route acquisition time**

The time required by a node to establish a route when requested. This measure of delay is especially of interest in on demand routing designs, but also finds applicability in table driven protocols through computational time.

- **Percentage of out of order delivery**

Not only must the amount of data packets dropped be taken into account, but also the time in which they are received. This issue of synchronisation becomes of great importance when in order delivery of information is expected.

- **Efficiency**

Questions the way in which the solution to a problem is reached. This takes into account the limiting factors that are experienced in an

⁴ Request For Comments. An RFC is a standard document describing protocols, systems, or procedures used by the Internet community. For example, the IP network protocol is detailed in an RFC (RFC 791), as are many others. All Internet standard protocols are written up as RFCs.

environment of high mobility, such as power dissipation, bandwidth, continuous link changes, etc. An example of this can be seen on control and data traffic sharing the same channel; excessive control packets will drain the performance of the network.

Each of the above metrics has a degree of generality arising mainly from it been in a document requesting comments. The authors of [77] leave it to the regression of the reader to use their own metrics, provided that a comparison with the performance of other protocols is made. For the metric of efficiency, they propose the following tracking ratios, stating that there are others which have not being considered:

- **Average number of data bits transmitted over the average number of data bits delivered:** Relates to the bit transmission efficiency on the network, as well as the average hop count taken by data packets.
- **Average number of control bits transmitted over the average number of data bits delivered:** Measures the bit efficiency of the protocol in disbursing control overhead to deliver data.
- **Average number of data packets transmitted over the average number of data packets delivered:** Measure that tries to assess the efficiency towards channel utilisation in terms of a bit count ratio.

From the above, we can see that a number of ways of assessing the performance or routing protocols exist. As in the case of efficiency metrics, each one can become more specific depending on the protocol design as well as the way in which the MANET is configured. Parameters that should be varied include the

number of MNs, the average connectivity of each one, as well as traffic patterns experienced among others. Consistent comparative results while varying each of these parameters, does yield a successful protocol that can be used outside simulation scenarios in future ad hoc networks.

2.10 Conclusions

This chapter has presented the general characteristics possessed by wireless routing protocols. Further to this, a number of designs have also been studied, as well as methods of comparing between different protocols. Having seen the different frameworks of wireless routing operation, our main objective has been to familiarise the reader with the key concepts and tradeoffs that distinct protocols offer.

In pioneering a protocol from scratch, our first consideration goes towards the characteristics that enable us to classify it. As much as these characteristics are not set, their degree of flexibility cross-correlates with others; as an example, a cluster based protocol would not be able to function effectively if its operation was solely table driven. The two key ways in which this can be seen is through simulation results with respect to other protocols and the complexity involved in its operation.

From the methods of comparison studied, protocol complexity yields a theoretical upper bound with respect to the network, single based characteristics present pitfalls that should be avoided, while the typical characteristics presented in the IETF request for comment form the most solid technique.

Perhaps the most important factor worth taking away from this chapter lies in the fact that in order to understand the performance of a protocol, a simulation environment of mobile nodes attempting to exchange information should be utilised. This should involve practical scenarios involving the exchange and loss of information within a network. Benchmark results from other protocols should also be taken into account based on the typical characteristics utilised. For each characteristic, the way in which each it has been selected should also be considered; this has a high degree of importance especially in the cases where two or more exceedingly different designs have to be compared.

Finally, having presented wireless routing protocols, the next chapter focuses on the security that different protocols have to offer. This directly links to malicious behaviour and how that can affect any exchange of information. Ultimately, security in wireless routing will provide us with a different way of measuring protocol performance to thus compare different designs. Such an attribute is required for establishing the necessary feedback for an emergent system that automates this design process.

Security in wireless routing protocols

3.1	The elements of security in routing protocols
3.1.1	The computer element
3.1.2	The human element
3.2	Consequences of compromise
3.3	Classification of a malicious attack
3.3.1	Attack definitions
3.3.2	Classifying adversarial behaviour
3.3.3	Active attacker hierarchy
3.4	Protocol attacks
3.5	Security mechanisms and defence techniques
3.5.1	Cryptography and authentication
3.5.1.1	Basic terminology
3.5.1.2	Symmetric key cryptography
3.5.1.3	Public key cryptography
3.5.1.4	Digital signature algorithms
3.5.1.5	Hash functions and Message Authentication Codes
3.5.2	Intrusion detection
3.5.2.1	Solution tangents
3.5.2.2	Protocol correctness
3.6	Protocols incorporating security
3.6.1	The Secure Routing Protocol
3.6.2	Ariadne
3.7	Conclusions

This chapter focuses on the constituent elements of security in wireless routing. Taking a systems approach, the first section identifies the different elements and sub-elements within routing protocols from a security perspective. This is followed by the consequences of compromise and the different classifications of a malicious attack. Next, the attack definitions given are applied to the protocols studied in chapter 2. The second part of this chapter examines techniques of countering malicious intention within a MANET. For this, a number of cryptographic primitives are presented, as well as ways of understanding the occurrence of malicious intent across the network. Chapter 3 concludes with a brief review (aiming to mainly illustrate the applicability of the concepts presented) of two secure protocol designs, namely the Secure Routing Protocol and Ariadne.

3.1 The elements of security in routing protocols

Information and system security are multi-faceted disciplines [78] often presenting diverse and conflicting objectives⁵. In its broadest definition, the concept of security carries the freedom or protection from a particular danger or worry. As much as section 1.4 defines the context in which we use the term, it is important to question what the constituent elements of a MANET from a security perspective are. In order to achieve this, we first look at the objective of the system at hand and how it correlates with its functionality.

Ultimately, any network aims to allow for the exchange of information among users. Since in this case the system is a network of mobile devices, the maintenance of the attributes presented in table 1.2 (that represent data security in communications), demarcate this objective of exchanging information within the ad hoc networking environment.

As a derivative, the two main elements involved in the communication process are the portable devices acting as nodes and the users operating them. Even though these two underlying elements exist in virtually all scenarios involving computer security, as seen in sections 1.3 and 1.4, in the case of the MANET their relation goes beyond conventional levels. Consequently, the actions of these two elements reflect on the security of the network, acting as a system in the process of any communication taking place.

⁵ *“If you want total security, go to prison. There you're fed, clothed, given medical care and so on. The only thing lacking... is freedom.”* Dwight D. Eisenhower [1890 – 1969]

In security [79, 80], you are only as strong as your weakest link⁶. Thus the actions from either the human or computer side within the MANET will always reflect on the weakest link of the system. The reason behind security not being a single visited topic among researchers in ad hoc networking lies in the trade-off between user interoperability and malicious intent. Nodes need to cooperate in order to allow for any communication to take place and this cooperation poses the main vulnerability of the system. It is exactly this trade-off that outlines the weakest link within the MANET.

Trying to balance this trade-off within the system, the wireless protocol is responsible for the exchange of information taking place. As a result, it can be argued that the weakest link to security in ad hoc networking is reflected upon the routing structure deployed within the protocol. This protocol definition can be exploited both from the human, as well as the computer, element of the system. Such deductive logic illustrates the importance of deploying a protocol that does not leave room for manipulation from either side. The two subsections that follow identify the factors affecting security within each of the two elements of the system.

⁶ This fundamental rule in security places its roots in the famous quote: "*The chain is no stronger than its weakest link*"; quoted in politics by Vladimir Lenin [81], in philosophy by William James [82], as well as in a number of applied fields as this example [83] illustrates.

3.1.1 The computer element

Within the device operating as a node, certain actions need to have effect in order to allow for any information exchange to take place. Having their origin in the protocol of communication, these actions reflect their operation in both the software as well as the hardware components of the MN. Consequently, ill-defined protocol implementations can leave room for software bugs and viruses as well as for manipulating hardware. As an attack is most likely to occur upon the weakest link of the system, the computer element of wireless routing security mulls over all three of its sub-elements, namely the protocol definition and how that is being utilised, the supporting software responsible for handling incoming traffic and the hardware that controls it. This is illustrated in figure 3.1, below.

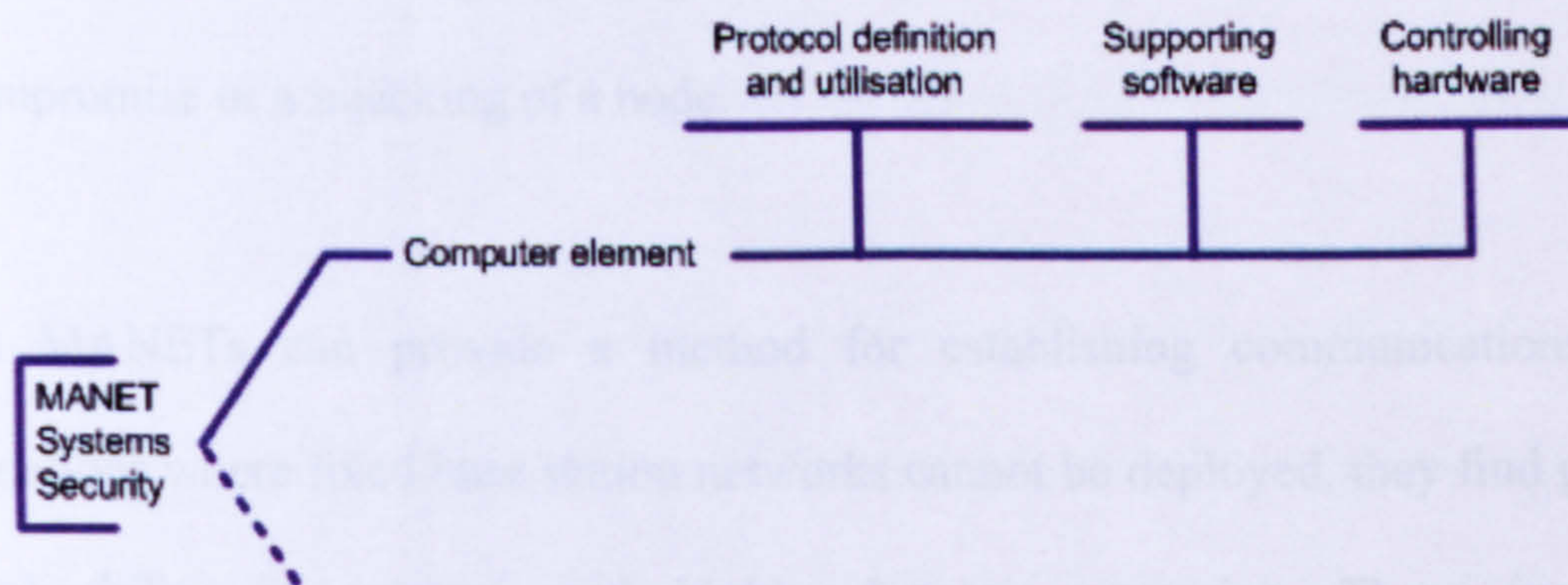


Figure 3.1: The sub-elements within the computer element of ad hoc network systems security.

Not only must the protocol safeguard its target operation by leaving no room for manipulation within its algorithm, but also both the software and hardware should be designed depriving any external factors from altering their original characteristics. Further to this, each of the three should allow in their specification the occurrence of misbehaviour from the other two parts and facilitate for a technique of detecting and reporting this incident.

From these, the hardware is the most secure sub-element, requiring physical tampering. Only recently have we seen in the area of software engineering the emerging field of secure programming [84 – 86]. In as much as ways of verifying routing protocols for MANETs exist [87 – 89], expanding their characteristics in a way that takes into account misbehaviour from the software or the hardware sub-elements is an area that remains to be studied further.

3.1.2 The human element

The human element within the system presents a much tougher adversary than the computer element in terms of security. No matter how well the machine is programmed and how well the interactions within the sub-elements of the device have been thought through, nothing can stop or prevent a physical compromise or a hijacking of a node.

As MANETs can provide a method for establishing communications in situations where fixed base station networks cannot be deployed, they find great applicability in extreme and highly dynamic scenarios. These include battlefields and other military applications as well as a number of emergency or recovery situations. As a result, the human element spans scenarios involving the capturing of one or more battle units in the battlefield, to search and rescue teams caught up in another aftermath of a tornado.

In computer security, it is often that we see the binary division of users with malicious intent and users with a healthy intent towards system. This is a characteristic frequently used in contemporary wireless security [90] allowing

anyone to promptly classify users, depending on their behaviour. The human element within any MANET expands this divide by making it more dynamic in time. Thus, any user that has in the past been considered a healthy user can, without prior notice, turn malicious towards any other user or group of users on the network. This process (illustrated in figure 3.2) summarises the devastating effects that the human element can have towards the security of the ad hoc networking environment.

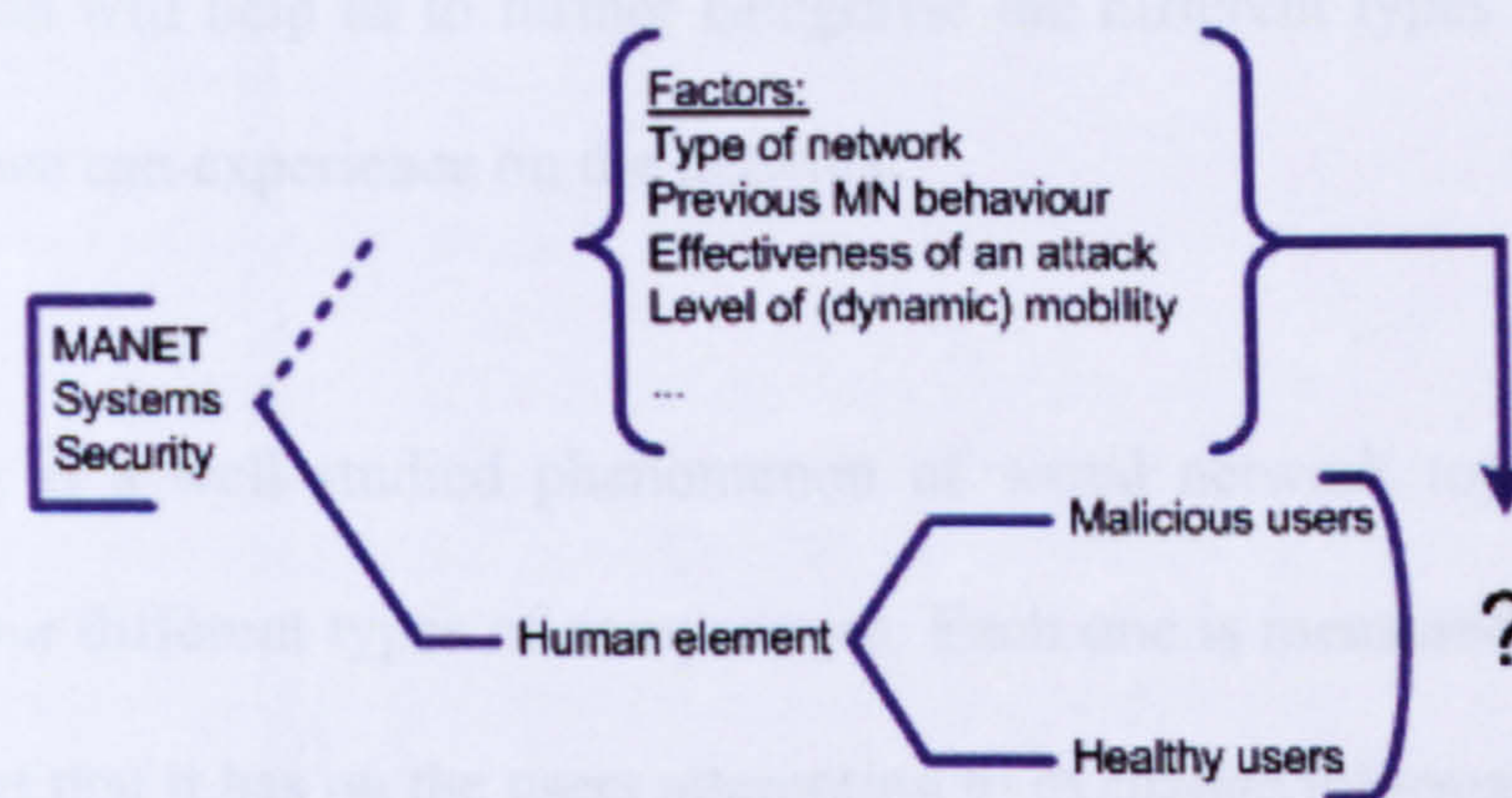


Figure 3.2: The decision making process within the human element and the main factors affecting it.

In metaphor to the human and computer element of security in MANETs, [10] presents two types of threats to routing protocols. The first one, inflicted by external attackers exploits injecting, replaying or distorting routing information as it is exchanged between nodes on the network. The second, described as more severe, comes from compromised nodes, which have in the past being healthy functioning nodes within the MANET. Even though each of these two threats share a number of common features identified within the respective computer and human element, it is virtually impossible to classify the cause of the threat to its origin. The important aspects that this classification does offer are the ability to further identify the types of attack as well as the consequences

arising from a compromise. The next two sections analyse these two subject areas.

3.2 Consequences of compromise

Having briefly seen the elements and sub-elements of system security in ad hoc networks, in this section we aim to provide the necessary motivation behind not allowing a compromise of the system to take place. The distinctions in this classification will help us to further categorise the different types of malicious behaviour we can experience on the network.

As routing is a well studied phenomenon of wired network topologies, [92] presents four different types of compromise. Each one is measured with respect to the effect that it has on the users attempting to exchange information. Since in mobile ad hoc networking, every node on the network also acts as a router, we consider the consequences of compromise in the wireless to, at least comprise of those seen the wired. As the following points present, the consequences of adversarial routers in a network can be disastrous. An attacker in control of any devices responsible for controlling the flow of data can:

- **Disable the entire network**

Bringing an entire network to a standstill is easily achievable if there is control over the method of forwarding data on it. Furthermore, any network outage is far more significant in a MANET, because recovering from it would require access to all the individual nodes that constitute it.

- **Use routers to attack internal systems**

As often networks interoperate with other networks, a compromised system can act as a foothold for attacking another computer or machine.

In today's standards, to prevent against this, it is not by chance that a wireless network often carries its own separate firewall.

- **Use routers to attack other locations**

Further to the point made above, an attacker can use one or more nodes to launch requests impersonating users from the currently compromised network. Thus, regardless of network interoperability, any external exchange of information confined within set protocol rules, can be done via impersonating other users.

- **Reroute all traffic entering and leaving the network**

The rerouting of traffic gives an attacker the ability to monitor, record, as well as modify any information exchange taking place. This raises a number of issues with respect to secrecy and data privacy.

From the above, we can see why the security of routing is vital to the existence of the network. Not only do all four of the above effects of compromise find common ground in the wireless but it also becomes apparent that recovering from such an attack is far more difficult in an environment of high mobility.

3.3 Classification of malicious attack

Following the classification of the security system which describes both computer and human elements within the MANET, this section provides a systematic description of the types of attacks that can be experienced in such a networking environment. The main objective of this section is to categorise an attack, depending on the actions that an adversary follows and regardless of the communication protocol in place. Having seen the different consequences from a compromise, this third and final section follows the security approach of

section 3.1, aiming to present the different adversarial models that can be witnessed within this system. For this purpose, we present every attack from a network perspective, taking into account the state of individual nodes, but not focusing solely on that.

3.3.1 Attack definitions

Prior to the categorisation, we proceed with defining the different types of attacks and the terminology accompanying them. This sub-section aims to provide the necessary terminology for the classifications of adversarial behaviour presented next; thus the focus of each definition resides with the definition of the attack and not the categorisation of it.

Wormhole attacks: Attacks which exploit the presence of a wormhole linking two or more nodes directly; a concept originally derived from the ‘wormhole’ singularities hypothesised in the general theory of relativity. In a wormhole attack [95 – 97] within a MANET, data packets are ‘tunnelled’ from one node on the network to another, thus disrupting the flow of information in time over the MNs of the network.

Replay attack: The retransmission of data packets on the network, having as an objective to actively disrupt the routing process. Such packets could be re-sent to nodes that have already received them with the objective of confusing the routing algorithm, or to nodes that are simply outside a particular routing process so that they cause a further disruption to network traffic. The origin of this attack resides within cryptographic protocols [98 – 100] and ways of attacking them and is also commonly addressed in secure e-mail [101].

Tunnelling attack: Similarly to a wormhole, a tunnelling attack exploits an existing tunnel between two or more nodes on the network. The difference between a tunnel and a wormhole [102] is that the wormhole can selectively tunnel any packets sent, but it does not have the capacity of exposing the origin of the tunnel, i.e. all packets from a wormhole come from its end node. This is not the case when a tunnelling attack occurs.

Node impersonation: Representing or imitating the behaviour of another node on the network, to the extent that fellow MNs believe that any data traffic they receive originates from the node at question. Impersonating attacks directly target the aspect of authentication [103] in the security of the protocol and the techniques that it utilises.

Libel attack: Incorporated within an attack taking place, the node(s) launching it aims to make other nodes believe that the origin of the attack does not point back to them, thus switching the blame to avoid any potential detection. These types of attacks present the justification for often adding non-repudiation to the list of attributes of data security in communications (table 1.2). Even though non-repudiation is useful for the detection and isolation of compromised nodes [10], it still remains a derivative of the three main attributes, and, as we shall see in chapter 5, is not required to be treated as a separate attribute.

Passive eavesdropping: The act of listening in to information being exchanged on the network, without modifying it in any way or form. For ad-hoc networks, passive eavesdropping does not necessarily imply a passive, receiving action from the node performing the eavesdrop; since every node acts also as a router, it could be the case that the node in question is required to forward the packets to another final destination.

Active attack: Attempting to alter the protocol in question to the advantage of the attacker. Generally, active attacks require active intervention [104]. The concept of an active attack resides as a superset of other attack definitions such as replay and rushing attacks, as well as packet injections.

Passive attack: Similarly to passive eavesdropping, an attack that does not involve the active interception and manipulation of data traffic on the network. Inversely to active attacks, passive cheaters follow the protocol, but attempt to obtain more information than the protocol intends them to [104].

Packet injection: A further description of the action taken by one or more nodes launching an active attack. This targets the consumption of resources on the network (such as power and bandwidth) requesting other nodes to handle data packets that do not form a part of the communication protocol being employed. It comes as a direct consequence to say that all active attacks use packet injections within the network.

Routing loop: A feature seen in DVR protocols within the classification of section 2.1 and the description of DSDV in sub-section 2.3.1. From a security perspective, a routing loop represents the manipulation of routing traffic from one or more nodes, having as an objective to cease the communication process. This is achieved by placing all communicating parties within a loop of sending and receiving information that does not assist them in establishing a route of communication. The occurrence of a routing loop is generally considered a weakness of the protocol being utilised.

Routing table overflow: This type of attack involves the convincing of nodes that non-existent routes actually exist, having as an objective overwhelm the tables that store the routing information within each node. Such an attack can have a direct effect on both processing as well as memory and bandwidth requirements.

Black hole problem: An attack whereby a node convinces fellow MNs to route data packets via itself and goes further to disengage from the communication process by dropping any packets that it receives. The malicious node(s) are said to form a black hole [105 – 109] on the network and the problem caused by such behaviour is addressed as the black hole problem.

Gray hole problem: A special type of black hole problem, in which the malicious node selectively drops incoming packets while forwarding others. This process confuses neighbouring nodes towards both identifying the node's malicious intent, as well as exchanging information on the network.

Denial of Service (DoS): An attack that targets the availability of information on the network. A Distributed DoS (DDoS) attack occurs when a number of compromised systems attack a single target, therefore causing a DoS for users on the targeted system. The flood of incoming packets causes the system not to be able to handle legitimate users, resulting in data inaccessibility. The specialities of a DoS attack within a MANET are two [108]. Firstly, an attacker could force a node to replay packets having as an objective to exhaust its energy [6], and secondly an attack can take place on all the layers of the network protocol specification being utilised [10].

Rushing attack: An attack which exploits the routing discovery process within on-demand routing protocols. A malicious node [105] advertises itself as holding a fresh route to the target MN that is inclusive of it in the hop path. The malicious node rushes to reply to the route request with this information making certain that its route is selected instead of another one. Consequently, having guaranteed that any outgoing traffic will pass through it, this node can choose to launch a variety of either passive or active attacks towards the incoming packets, regardless of knowing the whereabouts of the target node.

Byzantine behaviour: Any action by an authenticated node that results in disruption or degradation of the routing service [92]. This concept supersedes most of the active attacks described above, involving the interception, modification and fabrication of data packets on the network.

As it can be seen, most of the definitions above carry overlapping terms and descriptions involving similar attacks. Most of the terms in this section, either encompass or form components of other attack definitions also seen here. For this reason, we consider the classification of malicious behaviour on the network and go further to question where each of the attacks presented above reside in this classification. The following section aims to provide the necessary clarity for categorising the definitions presented.

3.3.2 Classifying adversarial behaviour

From the literature, two main ways of classifying adversarial behaviour in an ad hoc network exist. The first [92], captures and ranks the number of dimensions that each of the adversaries have at their disposal, while the second [93, 94], bases its approach on the classification of passive and active attacks.

From the two classifications, the latter represents nothing more than a simple expansion of the distinction between passive and active attacks. Commenting on such a method of comparison, we have seen a similar categorisation in subsection 2.9.2. The former uses a number of dimensions, in each of which a further ranking of the adversary is given based on the power and control that they have over the network. In deciding the dimensions, the authors of [92] argue that the different bounds on packet loss under each adversarial model are considered. Thus, the dimensions presented are:

- **Malignancy of the attacks**

An adversary is classified depending on their ability to passively or actively alter data packets across the network. Attacks within this dimension have a

wide range, from impersonation and libel (who to blame) attacks, to misrouting and topology distortion.

- **Collusion power of adversary**

Adversaries are classified depending on their joint knowledge, involvement and consent in a specific attack. The two limiting boundaries that specify the range of this dimension involve all malicious nodes acting on their own, to all adversaries conspiring as a team against healthy nodes on the network. Attacks within this dimension are mainly active; often wormholes and tunnels are used for the exchange of information between adversarial nodes.

- **Adaptivity / Intelligence of the adversary**

This dimension focuses on each node's ability to plan and adjust its behaviour within the network. The range of this dimension spans from completely static adversaries that attack the network in a premeditated way, to malicious nodes that will selectively attack MNs, depending on the traffic patterns they notice on the network.

Following the brief description of each of the above dimensions, we present a further analysis of the ranking within each one, attempting to identify which attack definitions are more relevant in each case.

For the dimension describing the malignancy of an attack, three main rankings (seen in table 3.1) exist. Starting from the weakest to the strongest, a benign adversary generally performs passive eavesdropping, having the option of operating as a black or a grey hole within the network. A static malicious adversary can launch a variety of active attacks, involving impersonation and

misrouting, as well as the attacks described above. Finally, a proactive malicious adversary carries the properties of both of the former, having the further ability to control the timing element of any active attack launched.

Table 3.1: The rankings within the dimension involving the malignancy of an attack, graded from the least (top) to the most (bottom) effective with regards to the consequences from a compromise.

Ranking	Description	Attack types
Benign adversary	<i>Cannot introduce information within the network, but can deprive information from other nodes within it</i>	Passive attacks, DoS, eavesdropping
Static malicious adversary	<i>Can inject as well as passively observe the information flow on the network</i>	Packet injections, replay attacks, node impersonation
Proactive malicious adversary	<i>Can dynamically select the most appropriate time for injecting information on the network, as well as perform all other attacks described</i>	Dynamic attacks, Byzantine attacks, rushing

For the dimension involving the collusion power of an adversary, two main distinct rankings are considered separate from the bound rankings of this dimension. In the simplest form, an attack coming from more than one node can involve no collusion from the MNs acting with malicious intent. Inversely, in the hardest possible case, it can be assumed that all adversarial nodes are fully cooperating with one another. The two rankings within these bounds represent the distinct cases of the number of nodes that are acting maliciously, as well as the level of cooperation that they might have between them.

Sub-section 3.3.3 that follows goes further to define an active attack depending on the number of nodes involved from the total in the network. An important aspect that disables us from producing a linear order to this dimension is the level of cooperation that malicious nodes will have between them. To question

this, we have to examine the motivation from any malicious attempt and whether or not it targets the disruption at a node or a network level. If the latter is assumed, the contradicting scenario of adversarial nodes turning against one another will unavoidably occur in time. Figure 3.3 defines the characteristics of the rankings within this dimension.

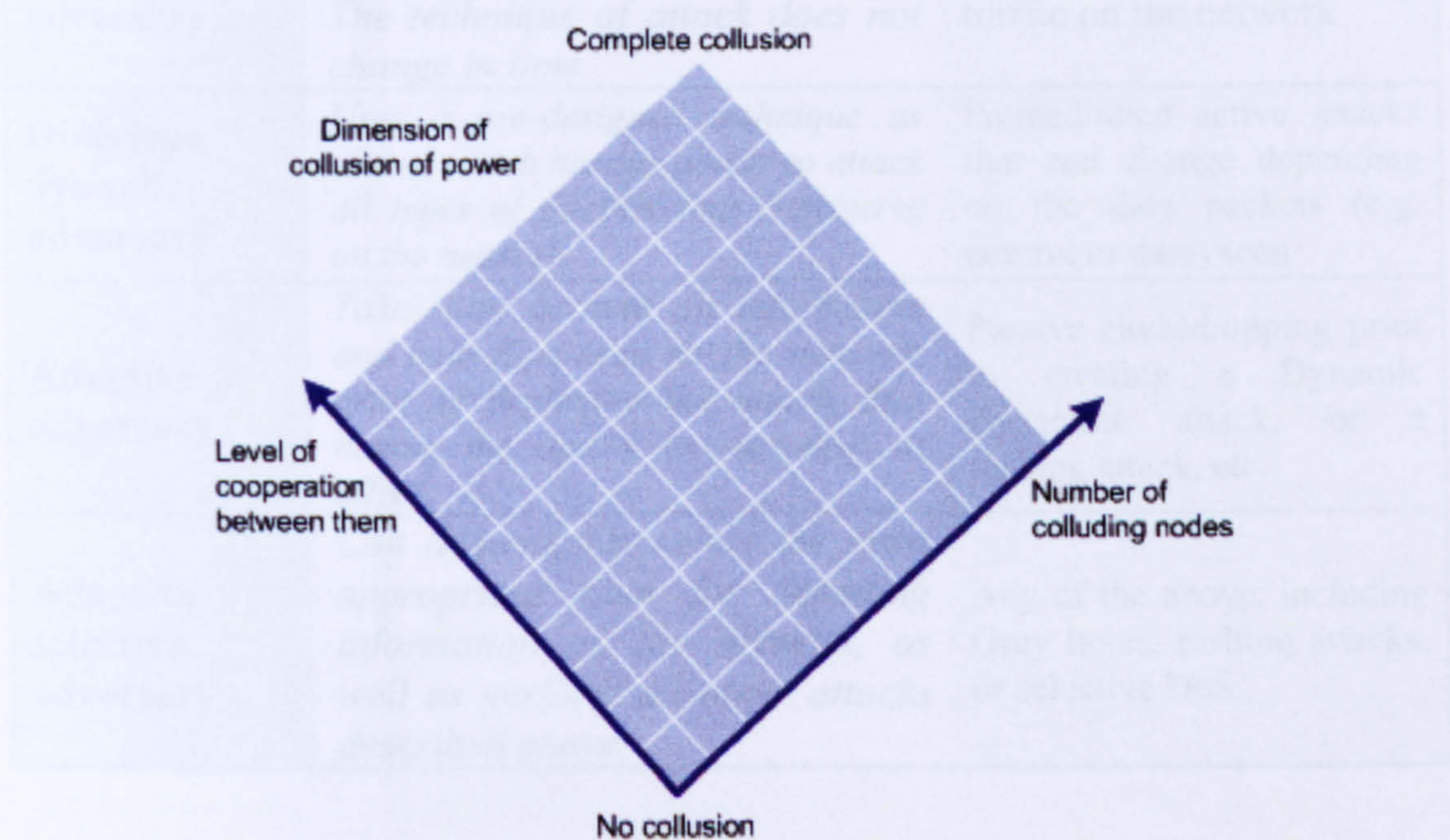


Figure 3.3: The categorisation of rankings within the dimension involving the collusion of power of adversaries.

For the dimension involving the adaptivity and intelligence of an adversary, four rankings are presented. These are dependant on the keenness of a MN to become accustomed to the communication processes of the network, spending the necessary time collecting information prior to deploying at attack. These rankings are described in table 3.2 below. In brief, starting from the least effective in terms of adaptation, an adversary is described as completely static, if the attack it deploys has a static (including any coin flips) fault pattern with respect to the routing algorithm. Such an attack involves no change in time and thus excludes rushing as well as passive attacks.

Table 3.2: The rankings within the dimension involving the adaptivity and intelligence of an adversary, graded from the least (top) to the most (bottom) effective with regards to the consequences from a compromise.

Ranking	Description	Attack types
Completely static adversary	<i>Uses a pre-designed technique of attack, while being completely aware of the routing algorithm. The technique of attack does not change in time</i>	Premeditated active attacks targeting specific traffic on the network
Oblivious dynamic adversary	<i>Uses a pre-designed technique as above, which has the ability to attack all types of packets that transverse on the network</i>	Premeditated active attacks that can change depending on the data packets (e.g. control or data) seen
Adaptive adversary	<i>Takes into account all information and coin flips seen on the network, prior to deploying the attack, thus having the ability to maximise its affect</i>	Passive eavesdropping prior to creating a Dynamic Byzantine attack, or a rushing attack, etc.
Adaptive selective adversary	<i>Can dynamically select the most appropriate time for injecting information on the network, as well as perform all other attacks described above</i>	Any of the above, including Grey holes, rushing attacks, or selective DoS

Following a linear model, the next two rankings, namely that of an oblivious dynamic adversary and an adaptive adversary incorporate adaptivity within the attacking fault pattern that they encompass. An oblivious dynamic adversary designs a dynamic fault pattern that can be adjusted depending on MN node behaviour, still remaining unaware of any non-deterministic (coin flips) elements of the algorithm. On the other hand, an adaptive adversary can not only adapt to the fault pattern but can also take into account any coin flips in real time, thus optimising an attack.

The ultimate ranking within this dimension, resides with an adaptive selective adversary, which further to the characteristics presented above, can also select the attacks depending on specific packet characteristics, such as origin, time-

stamps and so on. A specific and most effective attack in this case would be a grey hole, which despite of residing in the lowest ranking of the malignancy of attack dimension, can have the most damaging effects if performed in an adaptive manner.

3.3.3 Active attacker hierarchy

As shown in the previous two sub-sections, active attacks present a method of altering and manipulating traffic on the network, thus carrying a special weight in their effects. Consequently, a separate method for classifying them exists, which focuses solely on the proportion of malicious nodes that at any one time exist on the network. A point worth noting is that, in difference to the dimension involving the malignancy of an attack seen in table 3.1, the active attacker hierarchy presented in this section assumes that all actions originate from a single attacker, or at least from a group of adversarial nodes sharing common malicious interests on the network. It is this fact that makes this classification different to the dimension ranking presented above.

We define an attacker as an active $-n-m$ attacker, if they have compromised n nodes and own m nodes within the network. From this definition, the following measure surfaces, being directly linked to the security offered by the routing protocol. Figure 3.4 presents the different attacker hierarchies within the active $-n-m$ model with regards to the affect that they can have to a potential compromise.

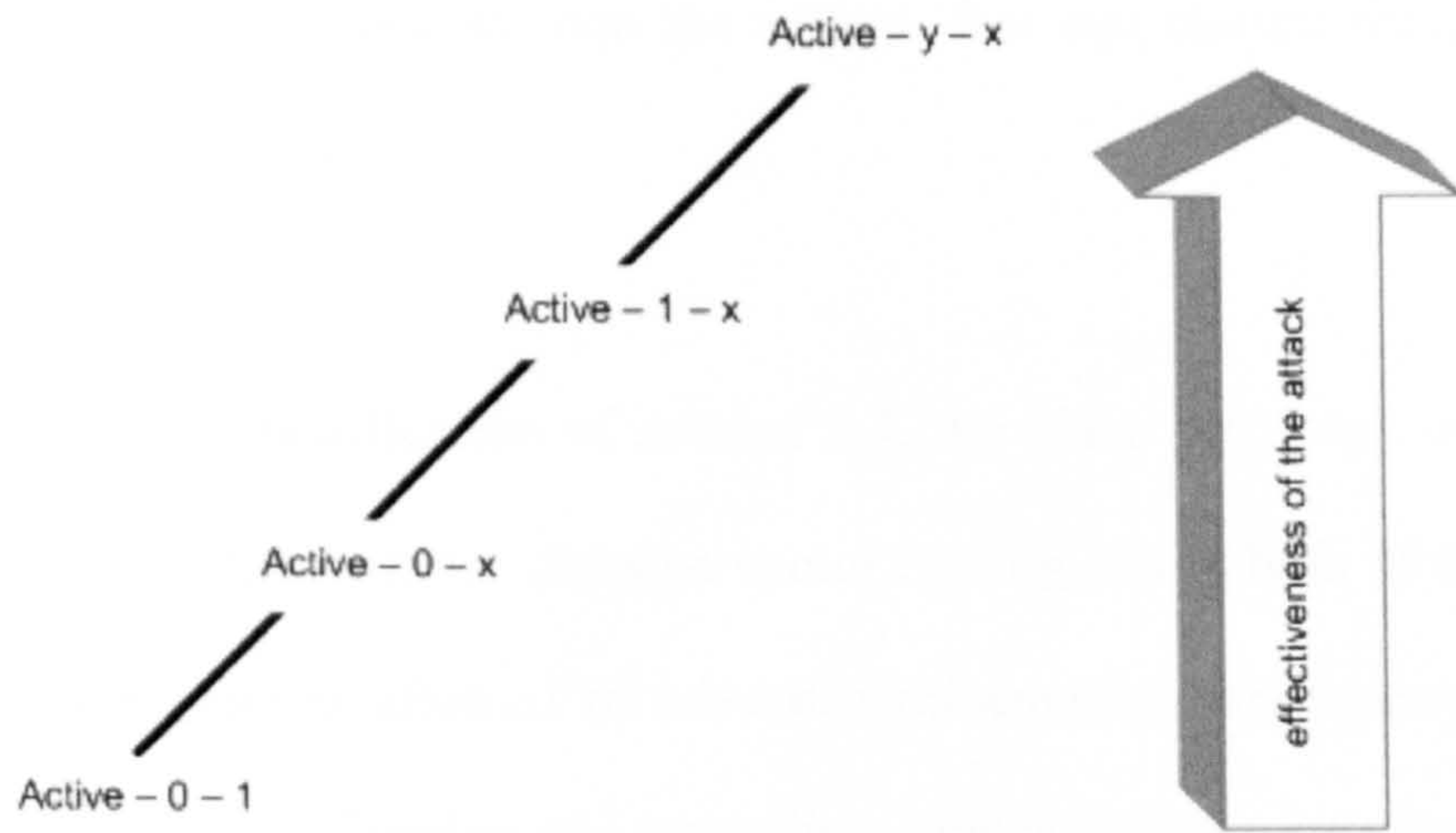


Figure 3.4: Attacker hierarchy using the active – n – m model, graded from the least (bottom) to the most (top) effective with regards to the consequences from a compromise.

This description seen in [31] makes the following distinctions. Starting from the least effective, an attack could vary from active – 0 – 1, thus assuming that no nodes have been compromised and that the attacker owns a single node, to active – 1 – x and active – y – x, where a number, y, of nodes has been compromised. In both cases, x represents that the attacker has a number of nodes under its control, different to the number of nodes that have been compromised.

3.4 Protocol attacks

Having presented the different classifications of a malicious attack, this section aims to put the above stated theoretical analysis to practical questioning, on existing protocols. Following the descriptions that this chapter has presented so far, we analyse a MANET as a system of security, aiming to illustrate different techniques of attacking known routing algorithms. For this, the protocol descriptions as well as their respective categorisations seen in chapter 2 are utilised. Moreover, extending from singular attack techniques, this section

provides a general perspective into the actions that can disrupt the process of exchanging information.

Starting from the classification of section 2.1, we place our focus on the first categorisation of link state vs. distance vector routing. As in both of these types of protocols a node is allowed to advertise information which they consider valid, a combination of replay and tunnelling attacks involving packet injection and perhaps the creation of routing loops could occur. Furthermore, black and grey holes could be created if one or more collaborating nodes advertise particular routes as optimal. In LSR, for such an attack to function correctly, a number of sequential updates promoting noticeable changes biased towards a single (malicious) route would have to flood the network. In the case of DVR for this type of an attack to function, there would have to be more than one MN collaborating with malicious intent, as there is a much slower convergence towards optimal routes.

For the categorisation of table-driven vs. on demand routing, a variety of Byzantine and active attacks could be used to target control information on the network. In the case of table-driven routing protocols, particular attacks could target table overflow, establishing routing loops as well as libel attacks aiming to shift the blame on other nodes, once the scavenging for the malicious MN has begun. For the case of on-demand routing protocols, the most interesting type of attack involves the occurrence of a black hole on the network. Attacks involving black or grey holes exploit the dynamic nature in which on-demand protocols establish their route and target the asynchronous element of the duration in

which a route remains valid. Consequently, protocols of this type, should not only describe the technique of establishing a route but also methods of safeguarding it against such malicious intent.

For protocols using periodical updates, active attacks by means of packet injections could lead to the temporary isolation of nodes, until the next scheduled update is due. This would be more difficult for event-driven updates, as an event (carrying data within the control packet information) is more likely to be noticed by more than one MN. Even though this might be the case, more complex packet injections, as well as DoS attacks could have twice the impact on the network, as they would impinge on not only the control but also the data information being exchanged.

In the category of protocols with a flat vs. a hierarchical structure, having a local central authority could prove quite resourceful in the event of an attack, as once that has been detected, it can be contained to the local cluster in which it occurred. On the other hand, an attack (either passive or active) launched by a cluster head would result in a complete standstill of all information being exchanged within that cluster. This would come as a direct result of the fact that the attacking MN would have all information available to manipulate, as it would be acting as a central authority.

Protocols using a distributed approach into computing routes are more vulnerable to active attacks targeting control information across the network. Even though this might be the case, decentralised computation of routes lowers

the level of awareness of neighbouring nodes, thus making it much more difficult for malicious users to be tracked down. Subsequently, neither of the two criteria within this classification secures the infrastructure as a standalone feature within a protocol description.

On the packet level of source routing vs. hop-by-hop routing, allowing the packet to carry redundant information regarding its origin and previous hop, despite being a burden for bandwidth and power dissipation requirements, does safeguard nodes from attacks. Thus, a rushing attack involving an ill-defined route could be ignored, provided that other route requests carry redundant information involving their journey across the network.

Finally, in the category of a single path vs. multiple paths, despite the redundancy that multiple path protocols might offer, having a single, but correct route does imply a smaller probability of a malicious MN interfering. Consequently, a trade-off exists between information passing through nodes that could have malicious intent, to obtaining a route from a single neighbouring node that could also be acting maliciously at that moment in time.

To summarise, all of the categories seen above are vulnerable to a number of attacks, mainly focusing on active packet injections. Despite each criterion within a category providing certain features which are either more or less volatile from a security perspective, its counterpart, even though meeting the opposite requirements towards that criterion has its own vulnerabilities. Having presented a number of techniques for attacking different protocols in the

categories in which they reside, the following section aims to provide us with the necessary resources for safeguarding communications in the mobile ad hoc networking environment.

3.5 Security mechanisms and defence techniques

To secure the MANET we must take into consideration the system definition described in section 3.1. This comprises of the two main (human and computer) elements and their corresponding sub-elements. As a result, the security of the system must be safeguarded at two levels. Firstly, on the traditional level of communication (involving the misuse of control and data traffic on the channel) and secondly, on the dynamic wireless level, involving all the special characteristics (see section 1.4) that a MANET embraces. Despite these two levels being inescapably interlinked, this categorisation provides the necessary clarity for identifying the defence techniques at our disposal.

For issues arising on the first level described above, we embrace modern cryptography and the corresponding schemes that it makes available. This includes symmetric and asymmetric encryption, the use of digital signatures and message digests, as well as key management schemes. Section 3.5.1 provides the necessary descriptions for the above terminology, hinting on its usage in both wired and wireless communications. For the second, we question techniques of detecting the occurrence of malicious activity on the network from nodes that had originally been considered as healthy. Labelling this as *intrusion detection*, section 3.5.2 questions the ability of individual nodes to detect (through the use of a corresponding protocol) adversarial behaviour within a MANET.

3.5.1 Cryptography and authentication

This section aims to provide a summary of concepts regarding cryptography and the ways in which they are used to secure communications. It does not present a complete view of all the aspects or sub-divides within the terms defined; for that we encourage the reader to view the references given.

We define as a cryptosystem, a set of algorithms responsible for disguising information so that only certain users of the system can see through the disguise. Cryptography is the science of creating, deploying and using a cryptosystem. Cryptanalysis is the science of breaking cryptosystems and thus being able to see through the disguise even when you are not supposed to. Finally, cryptology is the study of both cryptography and cryptanalysis.

One of the cornerstones of modern cryptology is a set of principles presented in 1883 by Kerckhoffs [110]. Most importantly, within this work the general principle that the security of a system should reside in the secrecy of the key and not of the algorithm, is presented. This is contrasted with the sometimes deplorable practice of “security by obscurity” [111], where the secrecy of the algorithm is assumed. Any work presented within this section takes for granted the principles presented by Kerckhoffs, avoiding the concept of information hiding and steganography⁷.

⁷ An excellent introduction on the subject matter, with references to all modern applications and theory can be found in [112].

3.5.1.1 Basic terminology

Following the definitions and terminology used in [113], we provide a list of basic terms that will be used throughout this section. These are presented in two categories, involving domains of operation and their respective transformations.

Encryption domains

- A denotes a finite set called the *alphabet of definition*. The most common alphabet definition is that of the binary alphabet, where $A = \{0,1\}$. Note that any alphabet can be encoded in terms of the binary alphabet.
- M denotes the set referred to as the *message space*, consisting of strings of symbols from the corresponding alphabet definition. An element of M is called a *plaintext message* or simply *plaintext*.
- C denotes the set referred to as the *ciphertext space*, also consisting of strings of symbols of an alphabet definition. The alphabet definition of C may differ from that of M . An element of C is called a *ciphertext message* or simply *ciphertext*.

Encryption and decryption transformations

- K denotes a set called the key space. An element of K is called a *key*.
- Each element $e \in K$ uniquely determines a bijection from M to C , denoted by E_e . E_e is called an *encryption function* or an *encryption transformation*.
- For each $d \in K$, D_d denotes a bijection from C to M (i.e. $D_d : C \rightarrow M$). D_d is called a *decryption function* or *decryption transformation*.

- The process of applying the function E_e to a message $m \in M$ is usually referred to as *encrypting* m .
- The process of applying the function D_d to a ciphertext c is usually referred to as *decrypting* c or the *decryption* of c .
- An encryption scheme consists of a set $\{E_e : e \in K\}$ of encryption transformations and a corresponding set $\{D_d : d \in K\}$ of decryption transformation with the property that for each $e \in K$ there is a unique key $d \in K$ such that $D_d = E_e^{-1}$; that is $D_d(E_e(m)) = m$ for all $m \in M$. An encryption scheme is sometimes referred to as a cipher.
- The keys $e \in K$ and $d \in K$ in the preceding definition are referred to as a key pair and sometimes denoted as (e, d) . Note that $e \in K$ could be equal to $d \in K$.
- For the construction of an encryption scheme, we are required to select a message space M , a ciphertext space C , a key space K , a set of encryption transformations $\{E_e : e \in K\}$ and a corresponding set of decryption transformations $\{D_d : d \in K\}$.

Following Kerckhoffs' principle, the sets $M, C, K, \{E_e : e \in K\}, \{D_d : d \in K\}$ are considered to be public knowledge. For two parties wanting to communicate securely using an encryption scheme, the only required thing that they must keep secret is the corresponding key pair (e, d) .

3.5.1.2 Symmetric key cryptography

The fundamental motivation underlying the applicability of a symmetric key algorithm, describes two parties, say, A and B, (a.k.a. Alice and Bob) wanting to communicate in the presence of a passive eavesdropper, say, E (a.k.a. Eve). To prevent Eve from understanding the conversation that Alice and Bob are having they use encryption, in a symmetric fashion, as shown in figure 3.5.

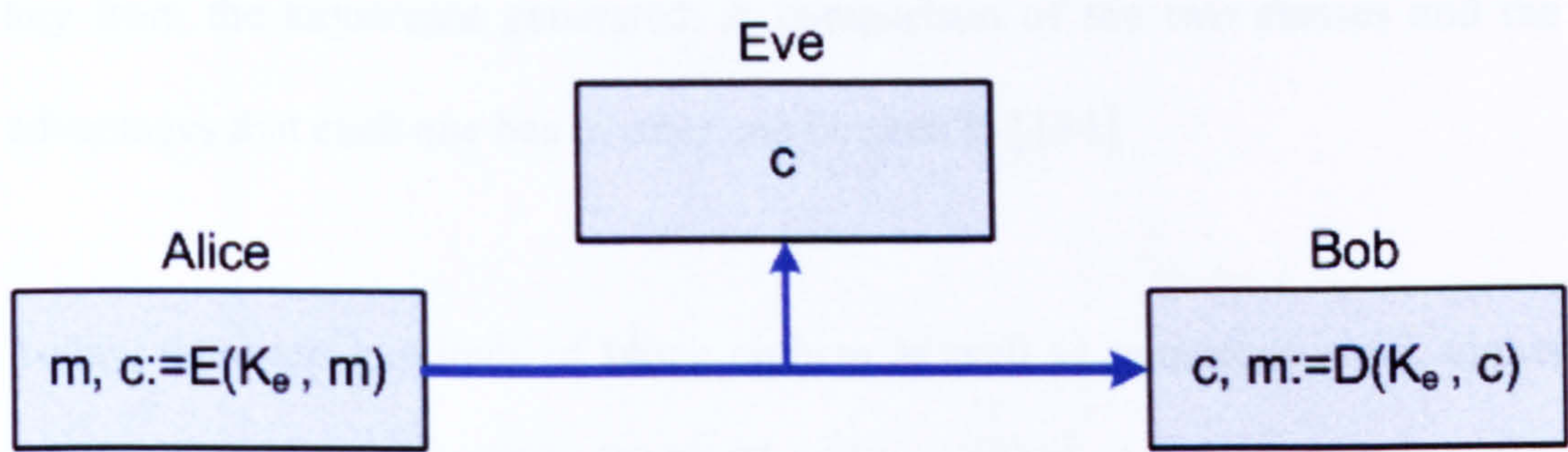


Figure 3.5: The principle of symmetric key cryptography and how it overcomes issues of passive eavesdropping.

When Alice wants to send a message m , she first encrypts it using the encryption function $E(K_e, m)$ of the pre-selected cipher that her and Bob have agreed on. Agreeing on the cipher is something that can be done over the public channel that Eve is listening into. What cannot be exchanged over the public channel is the secret key K_e , which both parties need to be sharing. For this, a separate communication channel that Eve cannot eavesdrop on must be used. As Alice transmits the ciphertext $c := E(K_e, m)$, Bob receives c and recovers the original plaintext m by using the decryption function $D(K_e, c)$, as that is defined by the cipher. Eve, not knowing the secret key K_e , is incapable of applying the decryption function on the ciphertext that she also receives.

Within symmetric key encryption schemes, two further classes are distinguished; block ciphers and stream ciphers [114, 115]. Block ciphers break up the plaintext message into a number of strings (called blocks) of a fixed length l , over an alphabet A , encrypting one block at a time. Stream ciphers on the other hand, operate by generating a sequence of keys (referred to as a *keystream*) and applying encryption on each plaintext string with the respective key from the keystream generated. A comparison of the two classes and the advantages that each one has to offer can be seen in [134].

Today, there are hundreds of block ciphers as well as numerous block cipher modes of operation available [116]. The first established, yet cracked, modern standard was the Data Encryption Standard (DES) [117], still partially in use today. The main upgrade after DES was 3DES [118], which performed three DES encryptions in a sequence. Ciphers designed within the last five years include the robust Serpent [119], Twofish [120], as well as the newly accredited Advanced Encryption Standard (AES) [121].

Similarly, a number of stream ciphers exist. As an example, we consider the RC4 stream cipher used in current wireless standards. The second implementation of the IEEE 802.11, namely 802.11b, describes a security protocol that utilises a form of encryption called Wired Equivalent Privacy (WEP). This protocol, despite of aiming to provide confidentiality, user authentication and data integrity in the wireless channel [122], fails to enforce all three of these requirements [123], due to the misinterpretation of some

cryptographic primitives [124], in the design. Weaknesses identified within RC4 [125] help increase further the vulnerabilities of 802.11b.

Utilising block ciphers within a MANET offers a unique way of safeguarding information against passive eavesdropping. The main disadvantage, apart from the obvious delay related to the encryption and decryption process, has to do with the fact that a key must exist for every pair of nodes wanting to exchange information. For n nodes this yields $\frac{n(n-1)}{2}$ key pairs. Adding to this, not only must a key be exchanged outside the conventional communication channel (in order to prevent eavesdropping), but also, the dynamic configuration of a MANET must be taken into account, where information is necessarily routed through other nodes in order to reach the specified target MN. Consequently, conventional problems with the exchange of the secret key [126] become far more difficult to tackle in an ad hoc networking environment.

3.5.1.3 Public key cryptography

One of the biggest problems that symmetric key cryptography is confronted with relates to the exchange of the secret key between Alice and Bob. Relying on the difficulty in solving certain mathematical problems (such as factoring), public key cryptography overcomes the limitation of having to share a secret key between two parties. Building on figure 3.6, Bob generates a pair of keys (S_B, P_B) one of which is private and one which is public. As a next step, he publishes his public key P_B , making certain to keep his private key S_B secret.

When Alice wants to send a message to Bob, she looks up the public key P_B , which Bob has published. With the help of the corresponding encryption function $E(K_{public}, m)$ that is part of the asymmetric cipher being utilised, Alice encrypts the message m using P_B . This yields the ciphertext c , which in turn is transmitted over the public channel. On the receiving side, Bob uses the corresponding asymmetric decryption function $D(K_{private}, c)$ to obtain the original message transmitted by Alice. Eve (as well as Alice and generally anyone apart from Bob) cannot decrypt the corresponding ciphertext c , as she is not in possession of the secret key S_B that Bob has.

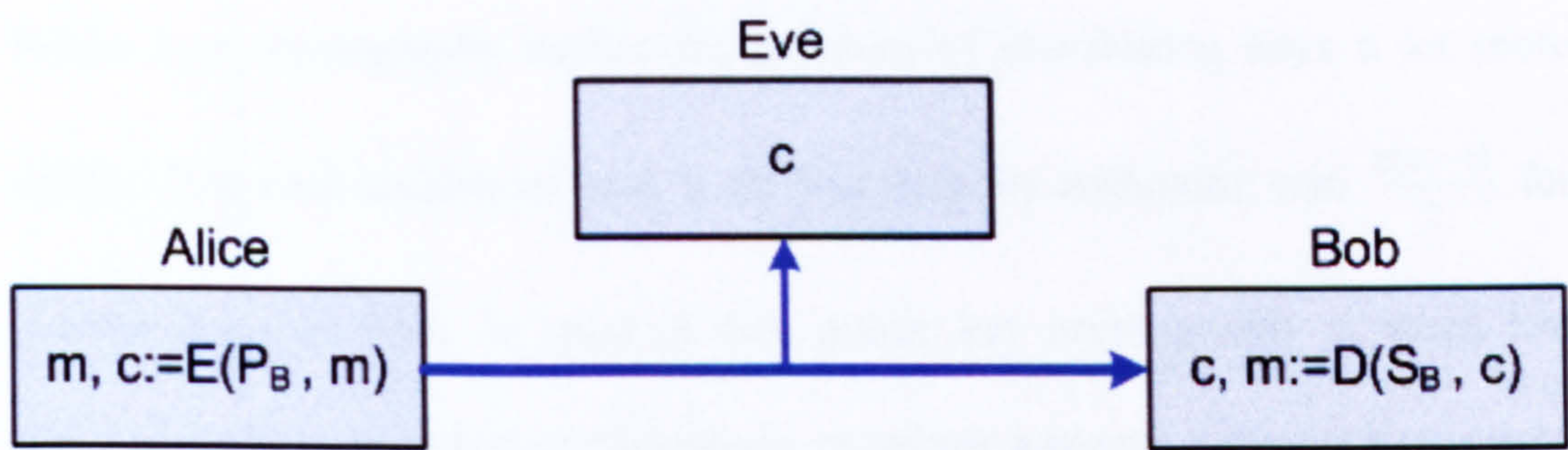


Figure 3.6: The principle of public key cryptography and how it overcomes issues of exchanging a secret key.

In order for asymmetric cryptography to function correctly, the key pair generation algorithm, encryption algorithm and decryption algorithm must ensure that the original message can be recovered. In mathematical terms, $D(S_B, E(P_B, m)) = m$ must hold for all messages m that the selected alphabet A can produce.

Two of the most popular asymmetric ciphers in use today are the Diffie-Hellman protocol [127 – 131] and the more widely used RSA public key cryptosystem [132, 133]. Public key cryptography was really started by

Whitfield Diffie and Martin Martin Hellman, with their 1976 publication of “New Directions in Cryptography” [127]. It describes a process for two parties so that they can agree on a secret key in such a way that both of them receive the same key without revealing it to someone that is listening to their conversation. RSA, on the other hand bases its operation on a one way function. Invented in 1978 [132] by Ronald Rivest, Adi Shamir and Leonard Adleman, it builds on the information required to invert the function used for the encryption process. As we shall see in sub-section 3.5.1.5, RSA can also be used for the generation of digital signatures.

Public key cryptography makes the problem of distributing keys a lot more simple. The total number of keys is far less than the arithmetic sum $\frac{n(n-1)}{2}$ for n nodes on a network. In spite of this, public key cryptography is much less efficient by several orders of magnitude, requiring a greater capacity in memory as well as processing time. For this reason public key cryptography is often used in conjunction with symmetric key cryptography ensuring the secure exchange of a key between the communicating parties.

3.5.1.4 Digital Signature Algorithms

Digital signatures provide a way of claiming the authenticity and integrity of any message, through the use of public key cryptography. In the generic setting, Alice creates a corresponding key pair (S_A, P_A) and publishes her public key P_A . Upon wanting to transmit a signed message m to Bob, she computes a signature $s := \sigma(S_A, m)$, through the use of the corresponding signature function of the cipher. Following this, Alice transmits m and s over the public channel to

Bob. This is illustrated in figure 3.7. On the receiving side, Bob uses the corresponding verification algorithm $u(P_A, m, s)$ of the cipher to compute the signature of the message m . For this, he uses Alice's public key P_A , the message m , as well as the signature s that he received. Provided the two signatures (the one computed with P_A and the one received) match⁸, the authenticity and integrity of the message are verified. If this is not the case, Eve is potentially attempting to impersonate Alice, or has tampered with the message that Alice has attempted to transmit to Bob.

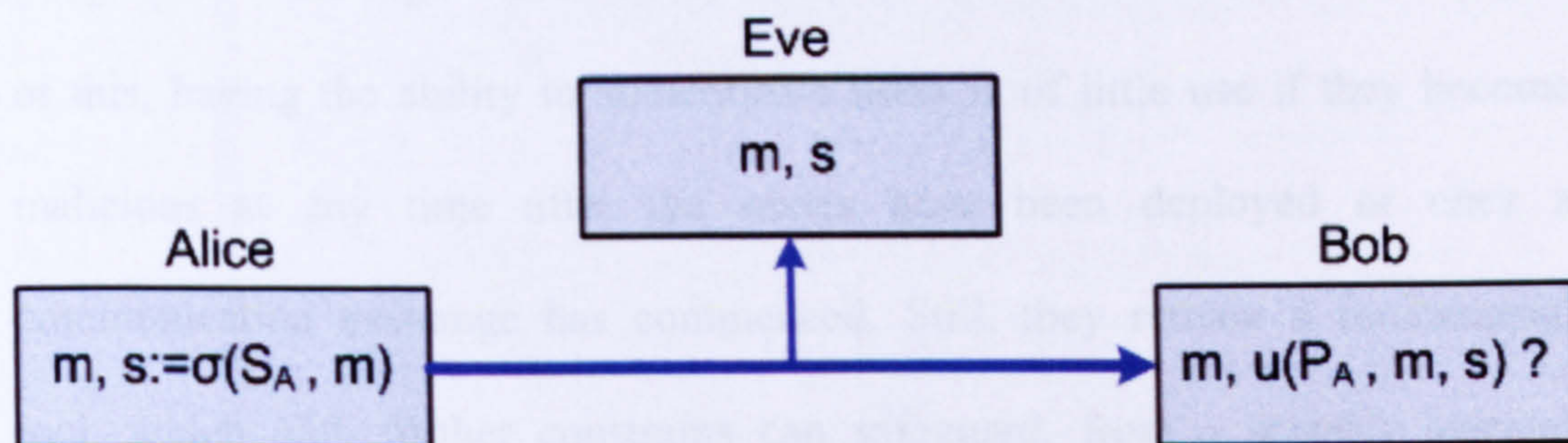


Figure 3.7: The principle of digital signatures through the use of public key cryptography.

The two main methods of authentication that satisfy the requirements to be used as a digital signature are RSA [132] and the Digital Signature Algorithm (DSA) [135] as it was published⁹ by the American National Institute of Standards and Technology (NIST) in 1994 [136]. The fundamental difference between the two is that even though they both safeguard message authentication and integrity,

⁸ By match we do not necessarily imply that they are equal. It could be that the cipher utilised specifies a 1 – 1 mapping between the signature generated using the secret key S_A and the one computed using the public key P_A .

⁹ Even though the publication, entitled the Digital Signature Standard (DSS), is mostly about DSA, it also approves RSA as a signature method.

DSA, unlike RSA cannot be used directly to provide message confidentiality. As Bob receives the incoming message with the corresponding signature, in RSA he recovers the original (signed) plaintext. On the other hand, in DSA he simply verifies if the message received is from the originating party (Alice) and has not being tampered with. Typically if DSA verification is successful, Bob accepts the incoming message, while rejecting it if that is not the case.

Digital signatures provide an excellent way of establishing the origin and integrity of a message within the dynamic environment of a MANET. Despite of this, having the ability to authenticate users is of little use if they become malicious at any time after the nodes have been deployed or once a communication exchange has commenced. Still, they remain a fundamental tool, which with further constrains can safeguard, from a security systems perspective, the elements that it was designed to.

3.5.1.5 Hash functions and Message Authentication Codes

Hash functions [137 – 140] often referred to as *message digests*, have as an objective to safeguard the integrity of the information being exchanged. A hash function takes as input an arbitrary long string of bits, generating a fixed-size result (typically measured in bits). The output of the function, referred to as the *hash*, is a shorter representation of the message. Hash functions find particular application in areas where performing an operation on the plaintext in its entirety is a costly option in terms of time and computation.

Given a message m , we can sign the message using one (or more) public key algorithms seen in the previous section. However, public key operations tend to

be fairly expensive in computational terms. Thus, instead of signing m itself, we apply the hash function h and sign the result $h(m)$.

In order to have the ability to represent a full message through its hash, two main properties must hold. Firstly, the function should be a one way function [141]; this implies that from the hash $h(m)$ it should be impossible to recover m . Secondly, the function must be resistant towards both strong and weak collisions [142, 143]. By weak collisions we imply that for a given hash of a message m_1 , it should be difficult (and ideally impossible) to find another message m_2 , for which $h(m_1) = h(m_2)$. By strong collisions we imply a one to one mapping; it should be difficult (and again ideally impossible) for two messages m_1 and m_2 to hash to same value, yielding again $h(m_1) = h(m_2)$. The above three requirements summarise the criteria that hash functions must meet.

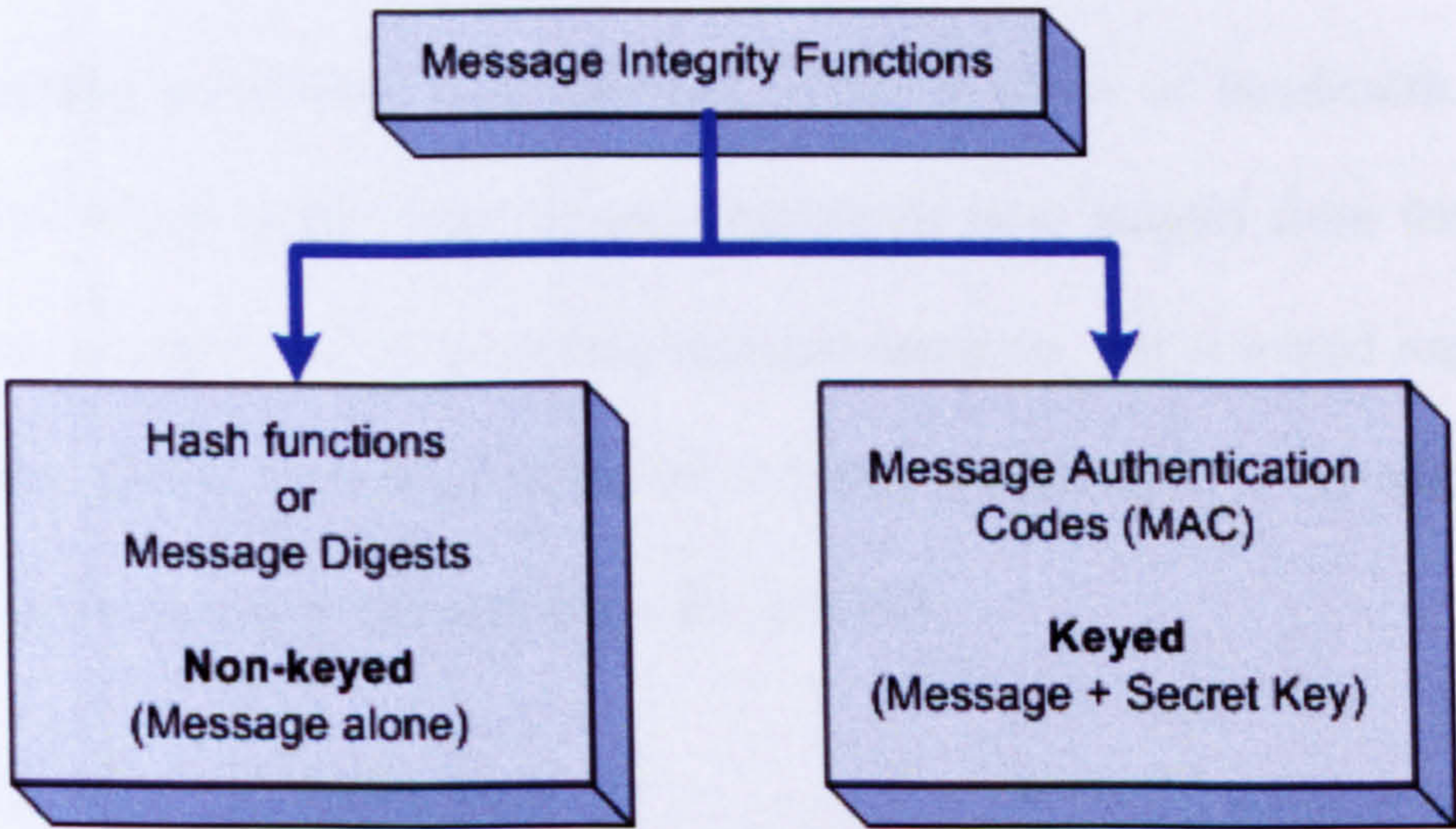


Figure 3.8: The categorisation of message integrity functions into hash functions and message authentication codes, depending on the use of a secret key.

Hash functions do not take any input in generating the corresponding output. Similarly to hash functions, there exist Message Authentication Codes (MACs)

for which the hash is computed dependant on a secret key. Consequently, for two parties to check the integrity of a message, a pre-agreement on a secret key must exist. This categorisation is illustrated in figure 3.8.

Today, the most popular hash functions in wide use include the 128-bit hash function MD5, developed by Ron Rivest [144], as well as the Secure Hash Algorithm (SHA) in its revisited form, namely SHA-1 [145]. They both base their operation on the predecessor of MD5, which was MD4 [146]. SHA-1 is a 160-bit hash function, with recent draft standards for three new versions [147], namely SHA-256, SHA-384 and SHA-512, having the respective number of bits as output.

In deploying a hash function or a MAC for use within an ad hoc network, an immediate trade-off is established. Safeguarding the integrity of any packet sent is something achievable; now carrying a cost in terms of bandwidth. Using SHA-512 which is the most secure (because of hash length) from the above functions would certainly guarantee message integrity, but it would imply that for every packet sent an overload of 512 bits would have to be considered, affecting the available throughput on the network.

3.5.2 Intrusion detection

No matter what the level of cryptography within the MANET might be, if a MN that is part of the network decides to turn malicious towards neighbouring nodes, the authentication techniques described above are of little use. Due to this fact, designing a secure protocol for a MANET is a task that could often carry problems which may be impossible to solve [148]. The main issue arises

from our inability to distinguish between a compromised node and a healthy but slow one. As the topology is constantly changing, nodes are always left to deal with out of date routing information, which in turn they must update in order to succeed in any data exchange. Consequently, identifying malicious intention is something that needs to be achieved while holding partial and potentially out of date routing information about the state of the network.

In order to be able to safeguard from a potential attack, the behaviour of the attacker must be noticeably different from that of a normally behaving router [149]. This forms the basis for the deployment of Intrusion Detection Systems (IDSs) [150, 151], which aim to detect traffic anomalies, as well as misuse patterns within a networking environment.

3.5.2.1 Solution tangents

This section aims to present a number of approaches that have been presented in the literature regarding the problem of secure routing with direct applicability in the ad hoc networking environment.

To protect against external attacks, Siroir and Kent [152] propose the use of keyed one-way hash function with a windowed sequence number for securing end to end communications and the use of digital signatures for messages sent to multiple destinations.

In the context of Byzantine robustness, Perlman [153] questions how to protect routing information, analysing the feasibility of maintaining theoretical connectivity in a dynamically malicious environment. Kumar [154] recognises

the problem of compromised routers as a hard problem, providing no direct solution. Based on an approach of detecting inconsistency using redundant information to isolate malicious routers, other works [152, 155, 156] offer partial solutions to the problem at hand. Finally, an authentication architecture for use within a MANET is presented in [157].

For distributing the trust within nodes of the MANET a popular technique is threshold cryptography [158, 159]. A $(n, t+1)$ threshold cryptographic scheme allows n parties to share the ability to perform a primitive cryptographic operation (such as creating a digital signature) provided that $t+1$ parties perform this operation jointly. Even if t parties collude to the same operation, the cryptographic operation selected cannot be performed.

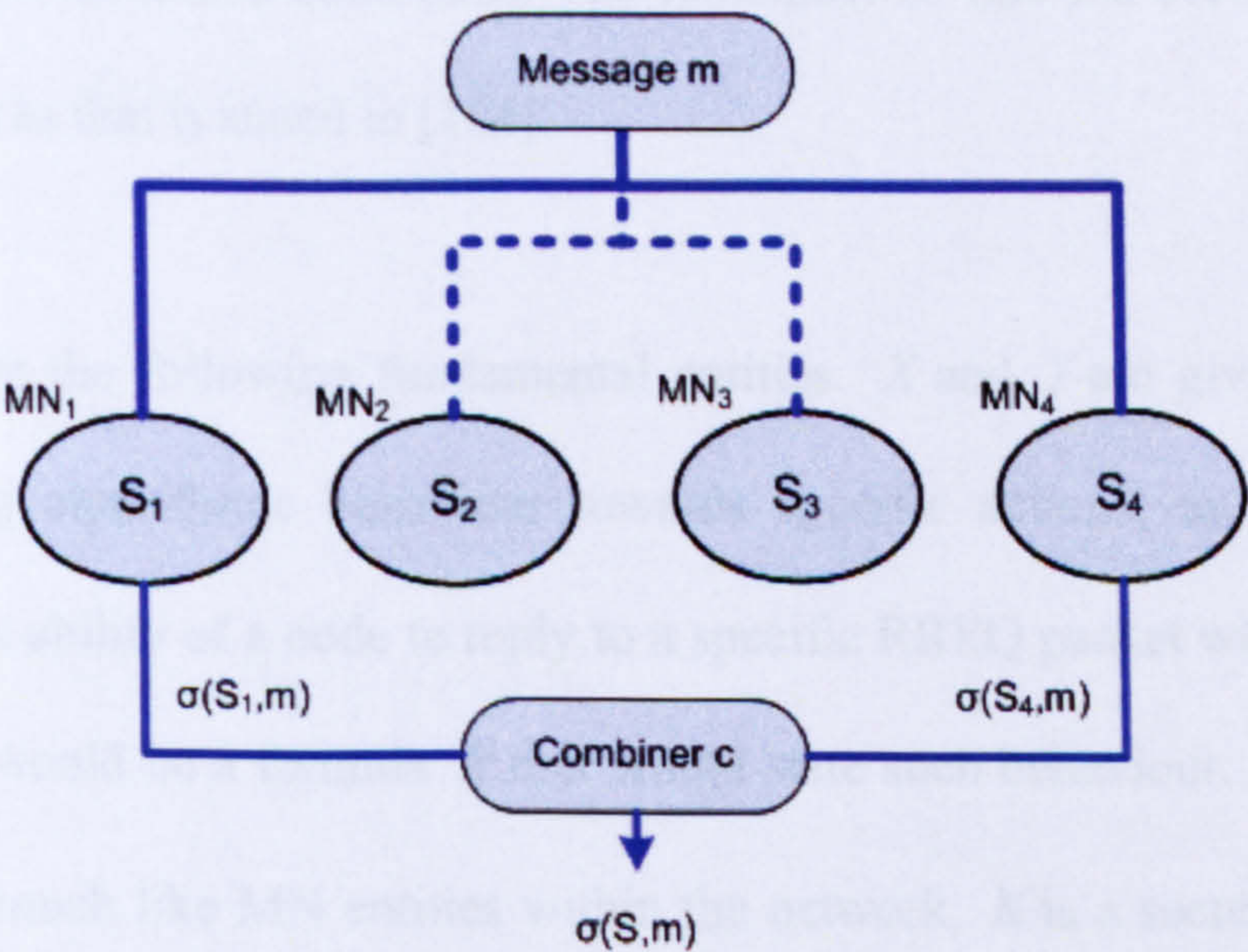


Figure 3.9: A $(4,2)$ threshold cryptographic scheme incorporating digital signatures; MNs 1 and 4 sign the message m with their partial key, thus allowing for the corresponding signature to be generated.

An example of this is illustrated in figure 3.9, where in order for the message m to be signed using the key S , at least two nodes have to contribute their partial keys (in this case S_1 and S_4), regardless of the behaviour of the remaining nodes.

Threshold cryptography offers the ability of partial agreement within cooperating nodes, which (provided t has been selected accordingly with respect to the total number of nodes n in their environment of operation) bypasses any potential occurrence of malicious intent. The applicability of this type of cryptography in MANETs and how it can protect against any compromised nodes can be seen in [160 – 162].

3.5.2.2 Protocol correctness

For presenting a formal analysis of a protocol and making sure that the stated design goals are achieved, the methodology seen in [163], following the inference rules presented in [164] a description of which follows. From these, our objective is to question the capability of one or more nodes, based on a number of premeditated conditions. The remainder of this sub-section presents the notation as that is stated in [164].

We consider the following fundamental entities. X and Y are given formulas representing algorithmic behaviour towards specific actions; as an example consider the ability of a node to reply to a specific RREQ packet while ignoring another. It would be a formula X that would state such behaviour. P and Q are principals, much like MN entities within the network. K is a secret shared key that P and Q share and C is a statement originating from one of the principals involved.

Within the above, the following notation is considered.

- (X, Y) represents the conjunction of the two formulas, which is treated as a set where commutative and associative properties hold
- $*X$ represents a not-originated-here formula property. If P is told X (see basic statements), it can distinguish it did not previously convey X in the current run
- $H(X)$ represents a one-way hash function of X

From this, a number of basic statements, using particular notation are considered:

- $P \triangleleft X$: P is told formula X
- $P \ni X$: P has or is capable of possessing formula X
- $P | \sim X$: P once conveyed formula X
- $P | \equiv \#(X)$: P believes or is entitled to believe that formula X is fresh
- $P | \equiv \varphi(X)$: P believes or is entitled to believe that formula X is recognisable; that is P has certain expectations regarding the contents of X before actually receiving it
- $P | \equiv P \xleftrightarrow{K} Q$: P believes or is entitled to believe that K is a suitable secret between P and Q
- C_1, C_2 : The conjunction of two statements, where commutative and associative properties hold
- $P | \equiv C$: P believes or is entitled to believe that statement C holds

- $\frac{P \triangleleft (X,Y)}{P \triangleleft X}$: A horizontal line separating two statements or conjunction of statements signifies that the upper statement implies the lower one. In this case, P being told formula set (X,Y) implies P knowing each of the formula's components and therefore X as well.

The above notation, as used for the protocol definition seen in sub-section 3.6.1, provides a method of authenticating protocol behaviour under a number of given assumptions and conditions. Consequently, the use of correctness proofs for wireless routing protocols acts as an initial benchmark towards testing the ability of the protocol to tackle the security problem under question.

3.6 Protocols incorporating security

Following the descriptions of the above techniques for securing wireless routing protocols, this section reviews two of the most popular secure protocol designs for MANETs. For each of the protocols, reference to the initial protocol characteristics that it encompasses is presented, followed by a description of the security mechanisms and defence techniques utilised by the specific design.

3.6.1 The Secure Routing Protocol

The Secure Routing Protocol (SRP) as presented in [165] can be applied in a multitude of existing routing protocol implementations. In particular, the designers of SRP suggest ZRP (section 2.6) and DSR (section 2.2) as two designs which can be extended in a natural way and with minimum modifications towards the packet structures travelling across the network.

The direct contribution of this protocol implementation is that it guarantees the ability for a querying MN to obtain correct topological information, despite the presence of malicious nodes. The one assumption that is made is that adversarial nodes are non-colluding, acting separately from one another.

For the SRP to function correctly, a Security Association (SA) in terms of a secret key is assumed between the source and destination node. For the exchange of this secret key, a number of methods (incorporating public key cryptography are available) [166]. Ultimately, we can assume that the secret key had been exchanged prior to the deployment of the MNs.

The source node S , initiates the route discovery process by constructing a RREQ packet containing a unique sequence number and a random query identifier. Along with the secret key $K_{S,T}$, the addresses of the source and destination nodes as well as the query identifier, act as an input to the MAC of the SRP. In addition, the addresses of the traversed intermediate nodes are padded onto the RREQ packet upon a new hop.

Provided that the data has not being tampered with, the request reaches the destination node T , which in turn constructs a route reply. It calculates the MAC of the route reply contents and returns the packet to S in a reverse hop fashion. In order to offer redundant information, the destination responds to one or more RREQ packets of the same query so that it can provide the source with as much information regarding the current network topology. Consequently, intermediate

(and potentially malicious) nodes do not have the ability to tamper with “hop data” as the MAC safeguards the contents of the reply from the destination.

As stated in [167, 168], the implementation of this protocol shows how impersonation and reply attacks can be prevented for on demand routing, by disabling route caching and providing end to end authentication through the use of a MAC cryptographic primitive. Thus, fabricated, compromised, or relayed route replies either never reach the route requester, or simply get dropped. An important note from this is that intermediate nodes do not need to be authenticated in order for them to have the ability to handle incoming traffic.

3.6.2 Ariadne

Another on-demand secure routing protocol that builds on the features of DSR (section 2.2) is Ariadne [94]. This protocol prevents attackers and compromised nodes from tampering with uncompromised routes, further inhibiting a number of DoS attacks. Additionally, Ariadne places a lot of weight on communication efficiency, burdening the implementation only with (less costly in terms of computation) symmetric cryptographic primitives.

The method deployed for authenticating routing messages utilises one of the following three schemes; shared secrets between each node pair, shared secrets between each communicating pair with broadcast authentication, or digital signatures. The designers of Ariadne place TESLA [169, 170] an efficient broadcast authentication scheme requiring loose time synchronisation, at the heart of their protocol. TESLA differs from traditional asymmetric cryptographic primitives, such as RSA, in that it achieves this asymmetry from

clock synchronisation and key disclosure instead of computationally expensive one-way trapdoor functions.

To use TESLA for authentication, each sender chooses a random initial key K_N and generates a one-way key chain by repeatedly computing the hash of the resulting value. Thence, $K_{N-1} = H(K_N), K_{N-2} = H(K_{N-1}), \dots$ etc. Each sender predetermines the time at which it publishes each key of its one-way key chain. This is done in the reverse order from that the keys were generated; in time, keys are revealed in the form of the sequence $K_0, K_1, \dots, K_N, \dots$ etc. As a result the authenticity of a message relates to the freshness of the key used in the computation and the ability to see which keys a sender may have already published.

Through this process Ariadne sets a number of design goals which provide resilience against active $-1-x$ as well as a number of active $-y-x$ attacks. Considering A and B to be principal MNs on the network, K_{AB} and K_{BA} to be the secret MAC keys between A and B , the $MAC(m)$ to be the MAC of the message m , using K_{AB} , the goals of Ariadne are as follows:

- A and B can authenticate the contents of both route requests as well as route replies
- B to have the ability of authenticating A and vice versa
- Any intermediate MN cannot remove an already present entry within a request or reply

In order to detect misbehaving nodes, Ariadne relies on a feedback reputation scheme, which notes the packets that have actually been delivered. Furthermore, this protocol incorporates malicious node avoidance in the route discovery process by including a list of malicious MNs and a corresponding MAC so that the list cannot be tampered with. Despite still having a number of weaknesses, this protocol represents a protocol design that bases its operation on an already established routing protocol, which is further expanded for the purpose of security.

3.7 Conclusions

This chapter has presented the subject of wireless routing security from a number of different perspectives. Firstly, we examined the MANET as system from a security perspective, incorporating a number of entangling elements and sub-elements. From this, and bearing in mind the consequences of a potential compromise on the routing level, we presented the different types of attacks, classifying adversarial behaviour on the network. The final aspect within the opening sections of this chapter questioned the applicability of a number of attacks that had already been defined on existing routing protocol designs.

Having identified the potential of a malicious MN (either alone or in a group) the second part of this chapter focused on methods for countering such malicious intention within an ad hoc networking environment. Despite the capabilities (in terms of exploitable vulnerabilities) that adversaries can have, we presented two protocol designs that incorporate a number of cryptographic primitives, thus managing to safeguard fundamental routing operations.

From the methods of attack and counter attack studied, we can see that MANETs are not in complete isolation from protection when it comes to securing the interoperability with which they function. Provided that the right tools are used in the design of a routing protocol, there is a tremendous space that remains unexplored towards the combination of fundamental cryptographic primitives in the rules for exchanging information.

Shifting away from wireless routing and more towards the contributing elements of this thesis, the chapter that follows examines the topics of emergence and adaptation in greater depth, having as an objective to utilise such concepts in the exploration of the design of secure protocol structures. Our motivation behind this comes as a direct consequence of the potential seen in the combining tools within cryptography for the security of an ad hoc network.

4.1	Introduction
4.1.1	Self-Organisation
4.1.2	Randomness, Complexity and Chaos
4.1.3	Swarm Intelligence and Stigmergy
4.2	Defining emergent behaviour
4.3	A novel system: Pandemonium
4.4	Adaptive systems modelling
4.5	Limitations of operation
4.6	Design pitfalls
4.7	Conclusions

In this chapter the concept of emergence is introduced, entailing the description of adaptive systems facilitating such behaviour. Firstly, a definition of emergence is presented; for this, the concept of self-organisation, as well as its relation to randomness, complexity and chaos is presented. Further to this, Swarm Intelligence and Stigmergy are reviewed as subject areas. As a next step, the Pandemonium system developed by Oliver Selfridge is presented. From this, we proceed into the description of the entities involved within adaptive systems modelling, identifying the limitations of operation as well as the design pitfalls that can be encountered in this modelling technique.

4.1 Introduction

Any formal discussion on the subject of emergence requires the active involvement of a number of areas. This section provides an introduction to these. Emergent properties are generally observed in highly complex systems that appear to hold an internal (non-linear) level of organisation. Within these, each element has a number of freedoms in their interactions, experiencing a degree of randomness that relates to chaotic effects. Such properties give rise to a holistic approach for the system that has states which cannot be premeditated. The terminology for each of the above is a science within itself; the sub-sections that follow offer the background required to study emergence, avoiding going into greater detail in each one. As notions such as complexity hold a number of definitions that could span into one or more chapters of this thesis, we settle for the corresponding references in each area and focus on their relationship with the field of emergence.

4.1.1 Self-Organisation

Commencing from our observation of the natural world [171], come a number of seemingly unrelated phenomena sharing several common behavioural characteristics that go beyond non-linear dynamics. These include, *inter alia*, Raup's data on the extinction of species [172, 173], Mandelbrot's analysis of commodity prices [174], as well as Gutenberg-Richter's analysis on earthquake magnitudes [175]. As suggested by the Bak-Sneppen model of evolution [176, 177], systems of such complexity are typically composed of a number of locally interacting components that operate at a critical state between chaos and order, which is known as Self-Organised Criticality (SOC) [178, 171].

The essence of self-organization lies within obtaining a system structure without explicit tampering or involvement external to the system. Elements of it have the ability to interact with other neighbouring elements. From these local interactions, a number of global characteristics for the system surface; in turn, these global features influence the rules between local interactions for all elements of the system. A popular example of this [171] is the image of a sand pile retaining its conical shape as more sand is added to it. As change comes through discrete events, each individual grain exercises a number of forces in its local neighbourhood that influence the dynamics of the system. Thus, as grains of sand drop into a sand pile, regardless of the speed or height of the drop, a conical mass of grains representing the whole system will be observed.

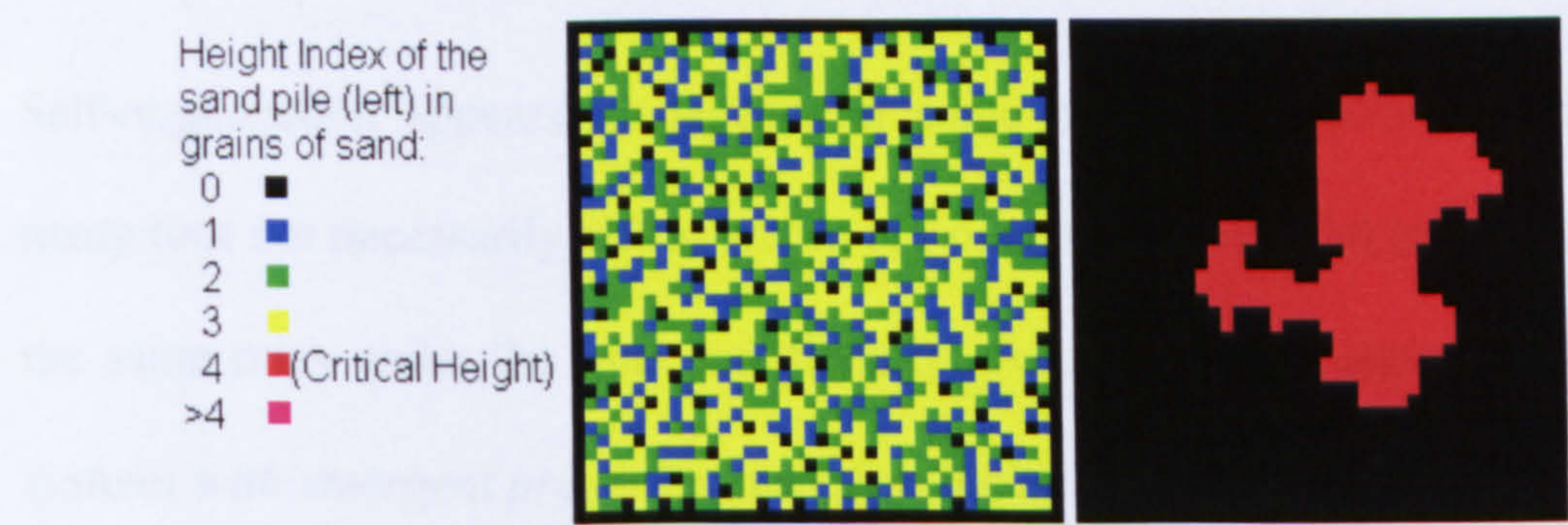


Figure 4.1: An instance in the simulation of a 2D sand pile model; the height of the sand pile in different locations (left) of the plane is represented by different colours. The right square shows the locations that toppled after the addition of the last grain of sand.

An example of a sand pile simulation is illustrated in figure 4.1 [179]. In this model, the sand pile is represented by a square lattice, where if there are 4 or more grains on any one location, the location topples. Toppling occurs by transporting the top grain to one of the 4 neighbouring locations, occurring

simultaneously throughout the whole lattice. The boundaries of the lattice are considered open.

Expanding on the above model, a number of theories of self-organisation in the context of physics and chemistry have been developed [180, 181]. Despite originally aiming to describe the emergence of macroscopic patterns out of processes and interactions defined at the microscopic level, these have proven useful towards understanding social insects¹⁰. From observation and as discussed in [182], self-organisation is indeed a major component in a vast range of collective phenomena within social insects. As a governing conclusion, it appears that complex collective behaviour may emerge from interactions among individuals that exhibit simple behaviour.

Self-organization appears to depend on some minimum number of entities, many (but not necessarily all) of whom are operating in parallel (i.e., following the same rules under the same conditions). As Flake [183] states: *“Complex systems with emergent properties are often highly parallel collections of similar units. Ant colonies owe much of their sophistication to the fact that they consist of many ants...A parallel system is inherently more efficient...Parallel systems that are redundant have fault tolerance...some ants die...similar ants can substitute...and subtle variation among the units of a parallel system allows for multiple problem solutions to be attempted simultaneously”*

¹⁰ As social insects we refer to insects living in colonies, such as ants, bees, termites and wasps , which are not solitary.

4.1.2 Randomness, Complexity and Chaos

Despite lacking a formal and complete definition, in the last 15 years the study of complex systems [184 – 189] has emerged as a recognised field in its own right. Perhaps the major drawback that deprives this field of a faster and more spread recognition lies in the inability of any definition of complexity [190] to quantify meaningful comparisons. Thus, statements of the form “*A is more complex than B*” are rendered trivial with respect to complexity.

An example of this is seen in Kolmogorov complexity [191 - 193] focusing on the shortest bit descriptions for any given Universal Turing Machine (UTM). In spite of the fact that, as stated in the Invariance Theorem [194], any two UTMs will agree on the complexity of a bit-string, say, x , there will always be two UTMs disagreeing on the complexity of two respective bit-string descriptions, say, x and y . As a direct consequence, even though in nature we have a number of species that are well defined through their genetic codes, any estimation of the complexity of different species will produce conflicting results.

Experienced within complex systems, emergence claims to be the key ingredient that *makes complex systems complex* [195]. As a concept, emergence outlines the occurrence of a new phenomenon that originally (in equilibrium) was not within the constituents of the system. How this occurs and whether or not emergence can truly take place within a formal agent-based system [196, 197] is a subject which at the moment is open to debate. Even though such questions are fairly new for artificial systems, the engineering of emergence has proven successful in a number of areas, such as the optimisation of the

performance of helical antennas [198], delivering promising results. The formal framework, linking emergence and complexity remains to be established.

Related work on the subject matter cites the following. Resnick [199] claims that: *“the study of self-organizing systems is, in some ways, the ‘related opposite’ of the study of chaos: in self-organizing systems, orderly patterns emerge out of lower-order randomness; in chaotic systems, unpredictable behaviour emerges out of lower-level deterministic rules”*. Holland [200], on the other hand, argues that complex systems are neither random nor chaotic, i.e., they are characterized by *“recognizable features and patterns”*. Finally, for Bak [171], complex systems are to be distinguished from both equilibrium systems and chaotic systems in which there is uniformity of behaviour (either very ordered or very disordered) throughout the system. *“Crystals and gases and orbiting planets,”* he suggests, *“are not complex, but landscapes are”*. Complexity for Bak is defined in terms of *“large variability”* indexed by scale-free phenomena in time and space (e.g., *“avalanches of all sizes”*, fractal structures) and that complexity so defined is an emergent characteristic of open systems that have self-organized to a state of *“criticality”*.

Emergence involves aggregate phenomena that cannot be intuited from the rules obeyed by singular entities. This too is a common refrain in the literature [171, 199, 200]. Again, citing Bak on the indicators of criticality: *“Zipf’s law as well as the other three phenomena (the power law, 1/f noise, and fractals) are emergent in the sense that they are not obvious consequences of the underlying dynamical rules”*. Hence, without the presence of a *ghost in the machine* [201],

emergence appears to unfold properties that would otherwise remain unnoticed, or considered to be of little importance, in some occasions treated as noise, small errors and the like.

4.1.3 Swarm Intelligence and stigmergy

Realising the great interest in designing systems with the ability to self-organise and even though the theory behind the subject matter is still evolving, has yielded the field of Swarm Intelligence (SI). The term SI was first used to describe cellular robotic systems [202–205], in which many simple agents occupy an environment and self-organise through nearest-neighbour interactions. Their objective was to generate patterns that typically relate to a specified task or design. In [206], this definition of SI was seen as “*unnecessary restrictive*” and was thus extended to “*any attempt to design algorithms or distributed problem-solving devices inspired by the collective behaviour of social insect colonies and other animal societies*”. Hence an area of practical interest within which emergence can exist as a documented phenomenon was established.

Within SI, emergence represents the ability of the system as a whole to be more than the sum of its corresponding parts (sub-section 1.5.2). As properties of elements within a complex system allow for more than one outcome when these elements interact, the behaviour of the whole cannot in principle be understood solely in terms of the properties of individual elements. In order though for the system to experience any behaviour at this level, the communication ability of neighbouring elements needs to be established.

This leads to the notion of stigmergy, first recognised and named by the French biologist P.P. Grassé (1959) during his studies of nest building in termites [207]. It describes the indirect communication in decentralised systems, in which the individual parts of the system communicate with one another by modifying their local environment. As this was first observed in nature, systems such as ant colonies are also referred to as stigmergic systems. A common example is nest-building in termites, which (like ants) also use pheromones to build very complex structures. For their interactions, termites like other social insects [208] rely on indirect communication mediated by modifications in their environment.

To summarise, SI represents the science of extracting design algorithms from natural systems, capable of tackling problems in a decentralised way and stemming from the collective behaviour of individual elements. These elements typically interact between one another via means of an indirect communication method. The process of indirect communication where elements modify their surrounding environment to achieve the exchange of information is referred to as stigmergy. As SI puts stigmergy to use, properties of individual elements that surface from the dynamics of interactions and which are new, are the result of emergence.

4.2 Defining emergent behaviour

Having presented the emergence from natural to artificial systems and the attempts to establish a formal theoretical background on the subject, this section defines the notion. The motivation behind a more formal framework comes from having the ability to be more precise concerning whether something is emergent or not. The definition presented within this section comes from Baas

[209]. Despite the criticism and expansions offered for this definition [210, 211], it yields a basis for understanding the occurrence of emergence in a system undergoing adaptation. Expanding on a theoretical perspective to avoid loss of generality is beyond the scope of this thesis. Our hypothesis is that in the process of problem solving for a given system, emergent properties surface that can be used by the system in the solution path. Thus the system can be seen as undergoing adaptation to achieve a desirable and improved solution state.

We begin by a system description of dynamically interacting elements. These elements we refer to as structures. Each structure is considered capable of carrying a number of properties. The only way to observe these properties is through an observer P of the system. The presence of an observer is central to the existence of emergence; to register that something new has surfaced, we need at least one mechanism for observing the corresponding structures. Consequently, we assume that emergent properties must be observable; but they appear because of the system of interactions among the structures, not because of our observation¹¹. This system description is presented in figure 4.2.

¹¹ Quantum effects between the observer and the system of structures, as well as the ability of the system to present more or fewer emergent properties depending on the level of observation are ideas that remain in the context of this footnote.

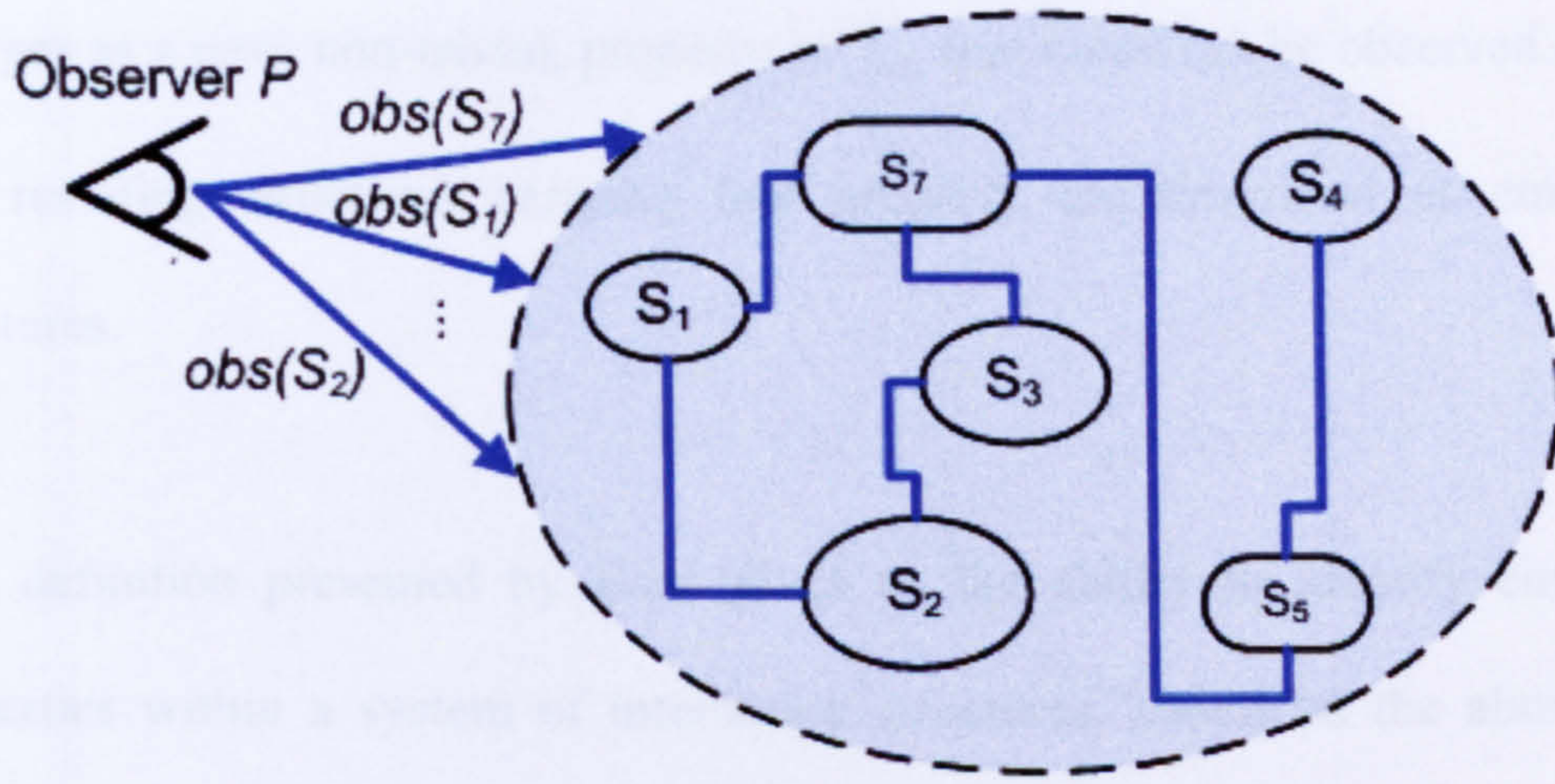


Figure 4.2: A system of finite interacting structures, under the observation of an observer P . Interactions are represented as links between structures.

The process of the emergence of properties on several levels may be considered as a result of a series of abstract construction processes, similar to mathematical constructions. Given a set, Σ , of first order structures, one can, by some kind of observational mechanism, $obs_1(\Sigma)$, obtain a 'measure' of the properties of the structures at this level. The set Σ can then be subjected to a family of interactions, using the properties registered under this observation.

2.1.3. Effect of the Definition of a Property on the Way of Constructing and Modelling

Assuming that the set of structures can dynamically change in time, we define the first instance in the observation as Σ_1 . Thence, the set of structures Σ_1 can carry a number of properties as those are defined by a function of observation $obs_1(\Sigma_1)$. From the interaction of these structures (involving an alteration in their property values through another function) a set of second order structures, Σ_2 , surfaces also carrying an observation function, $obs_2(\Sigma_2)$. At this point, a property E is defined as an emergent property of a structure in Σ_2 if and only if E is an element of $obs_2(\Sigma_2)$ but not an element of $obs_1(\Sigma_1)$. Consequently, E

emerges as a new, non-trivial, property in Σ_2 that could not be observed in Σ_1 .

The resulting structures carrying that property are described as emergent structures.

This definition presented by Baas gives us the ability to identify emergent properties within a system of interacting structures. Based on the above, the distinction and further categorisation between weak and strong emergence has also been made [212, 213]. Having presented ways of identifying it, the remainder of this chapter focuses on methods and techniques for the generation of emergence within a system. For this, we first look at Oliver Selfridge's Pandemonium model and then generalise that approach by presenting the techniques followed in adaptive systems modelling.

4.3 A novel system: Pandemonium

As presented so far, emergent phenomena are not just academic curiosities [214]. Given the definition of emergence, a way of designing and modelling such systems needs to be established. The earliest model to be developed taking into account such behaviour was "Pandemonium", a pattern recognition learning program developed by Oliver Selfridge that was (initially) used for detecting Morse code patterns. Selfridge in his 1959 paper [215] describes a decision making system involving four entities (he refers to them as demons), which upon receiving an input, through continuous interaction, are able to reach a verdict regarding what the input value actually is.

Even though the above description is no more than a basic neural network increasing its entropy in a way that is dependant on the input, the difference in

Selfridge’s approach lies in the fact that dedicated demons monitor the progress made. Demons decide, by a process of elimination, not only to keep specific candidate values but also how to improve successful candidates based on the findings of all the others. As a result, this second feedback loop leads to an adaptive system that, in spite of being governed by a set of very simple rules, has the ability to come to complicated (and correct) decisions.

Pandemonium consists of four distinct layers (figure 4.3 and figure 4.4), with each group of demons specialising in specific tasks. The bottom layer constitutes the storage area and hosts demons that store and pass on data. The second layer is composed of demons that filter and weigh the evidence from the third layer: This is where the information processing is taking place; computational demons process the data of the first layer into information and pass it on to the next level. Finally, at the top layer lies a single decision demon that decides what information was actually presented to the system. At first sight, this model appears to be linear; each layer processes information and passes it on to the next layer, but there is a lot more taking place.

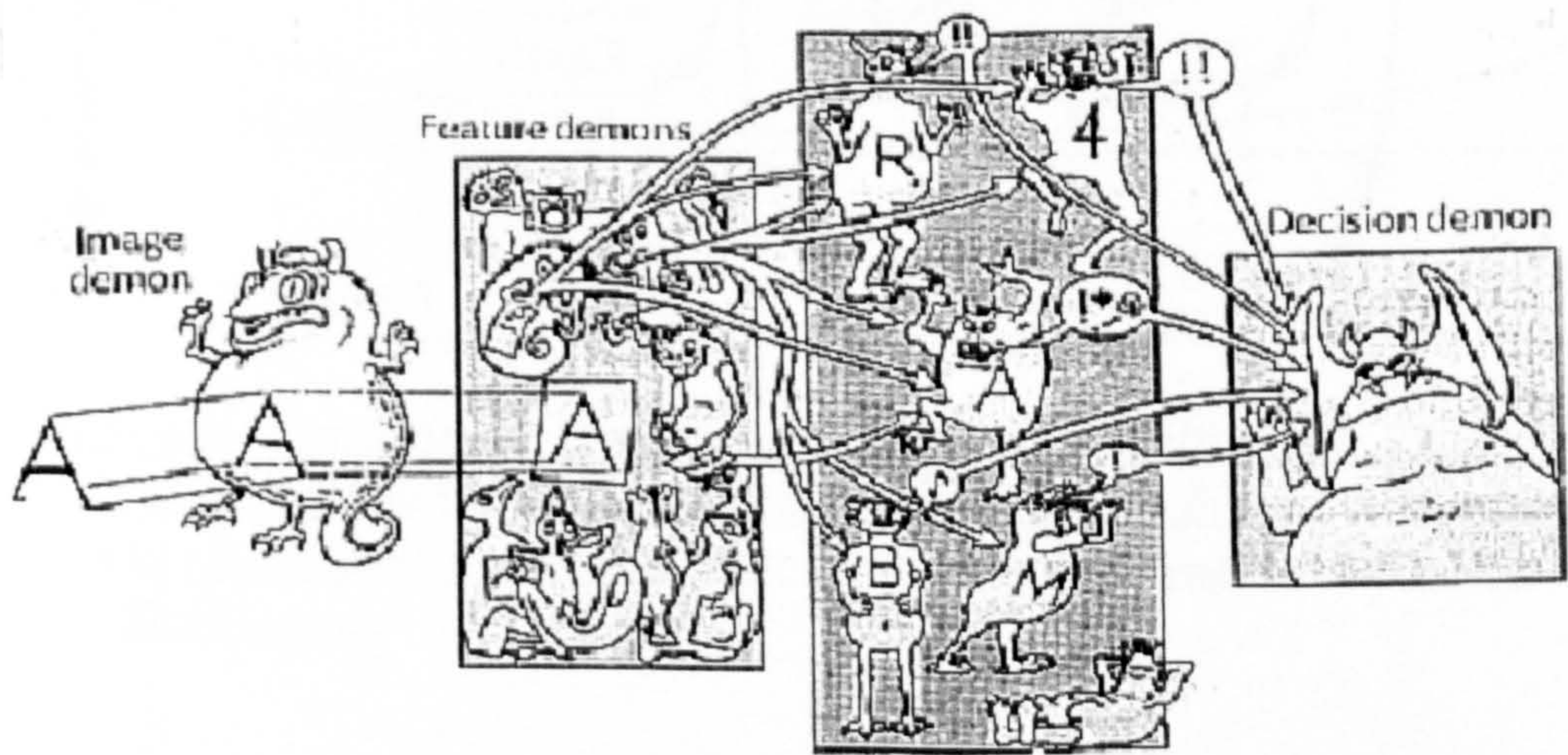


Figure 4.3: Selfridge’s Pandemonium model, as depicted in [216]. The interacting layers can be seen in boxes (excluding the image demon) attempting to make a decision on the character which they are identifying.

The uniqueness of Pandemonium lies in the second layer, which (as illustrated in figure 4.4) is made up from cognitive demons. These demons weigh the evidence produced by layer three and ‘shriek’ [217] the amount of evidence they have (or think they have) up to the top layer. The more evidence that is gathered, the louder the shrieking. At the top layer, information from layer three gets passed on; the decision demon also listens out for the loudest ‘shriek’ from layer two. This doesn’t imply that the decision demon is biased towards the loudest shriek; it simply listens to what other demons claim to have found of importance and then comes to its own conclusion regarding the output of the system.

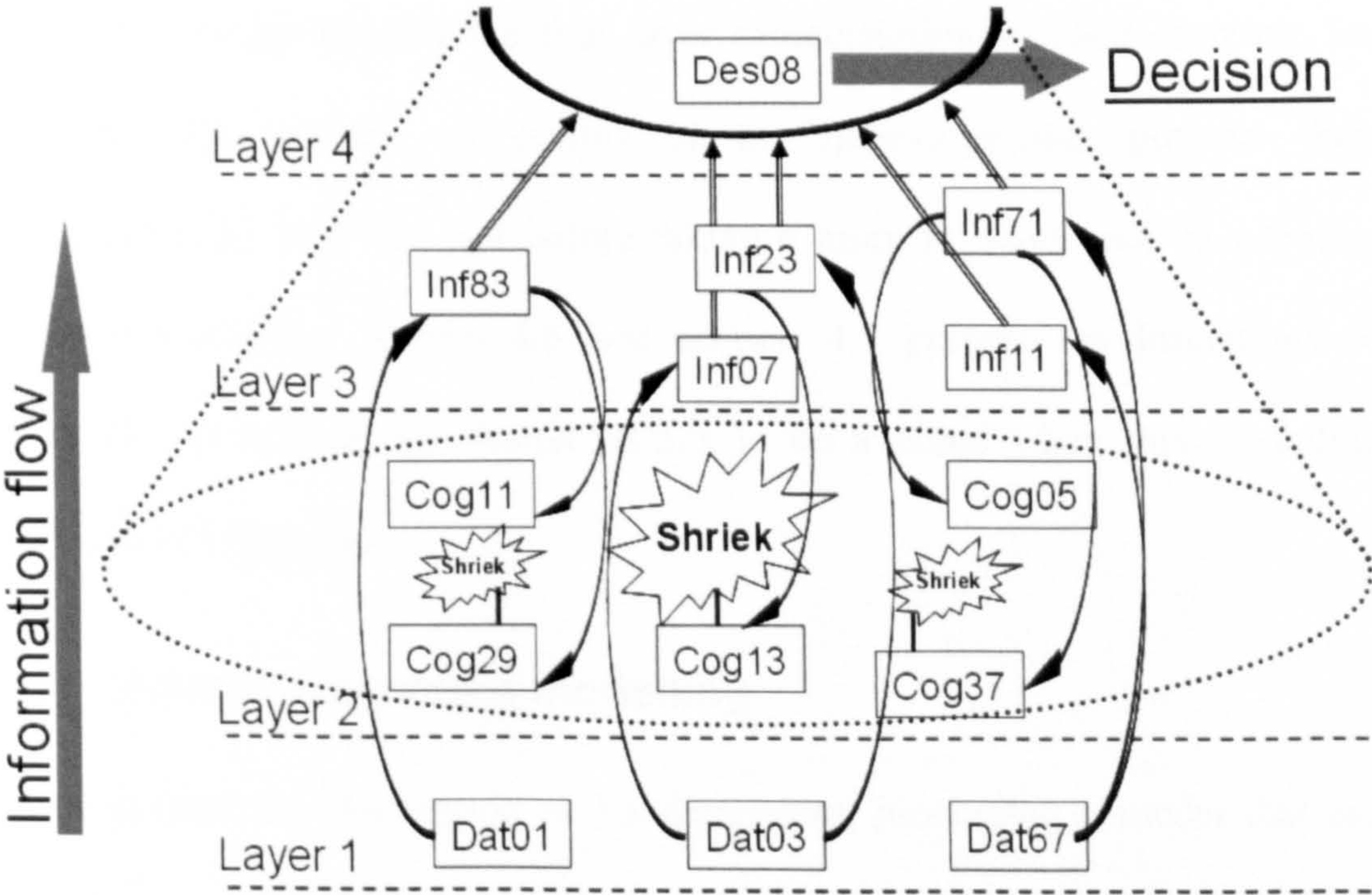


Figure 4.4: The abstract layered hierarchy within Selfridge’s Pandemonium model, exemplifying the interactions between layer 2 and layer 3. In order to be uniquely identifiable, demons of each layer carry a unique tag, shown as Dat67, Inf71, Des08, etc.

From this, if we imagine a pool of decision demons tackling the same problem, each backed by a self-governing Pandemonium, then, depending on the success

rate of specific ones, we can choose to disregard demons that have produced unsatisfactory results. Furthermore, by examining the inner workings of demons that have produced sufficient output, we can provide successful candidates with more knowledge regarding the solution of the problem, thus creating an ever expanding Pandemonium.

Hitherto, we have examined the detailed structure of an adaptive system that, given a problem definition, obeys a survival of the fit (and not fittest) scenario. That is to say, learn how to solve the problem, through a number of attempts, or die trying. The process of learning entails the occurrence of a number of emergent properties within the system model. In creating these demons, it is not enough to simply identify the four layer categorisation of Pandemonium; we have to also provide an outline of the behaviour that governs their communication. For this, not before taking a more in depth look in adaptive systems modelling, section 4.6 and section 4.7 present the limitations of operation, as well as the design pitfalls to be avoided when this modelling approach is followed.

4.4 Adaptive systems modelling

As seen from the description of Pandemonium, pioneering a model that can exhibit emergent behaviour goes beyond the conventional methods of analytical and simulation modelling. The system designed, would have to have the capability to generate new behaviour in an attempt to adapt to the problem definition at hand. Thus, a more useful way of categorising models, according to their purpose and amount of real-world complexity that they carry is used. This

classification scheme, proposed by Roughgarden [219], defines three categories of models, as described below:

- **Minimal models for an idea**

Aiming to capture the entire idea behind the system, minimal models encapsulate only the most fundamental parts, without incorporating a high level of detail. This is very useful in illustrating a general principle and stripping away extraneous information. Such models achieve a focus on the most relevant parts of the system, without neglecting others.

- **Minimal models for a system**

Without incorporating the finest level of detail and building on minimal models for an idea, these systems carry a greater amount of information. By representing generalised systems, they carry enough specificity to narrow the applicability of the model, defining it to greater detail than that of minimal models for an idea.

- **System models**

System models simulate actual systems carrying all the details and the descriptions that they can have. Typically, the first attempt in constructing system models involves the synthesis of different details into a global description. From that point onwards it is a matter of (often painstakingly) defining all the different attributes that the system carries.

The above three categories of models provide three different ways for emergent phenomena within artificial systems to surface. This categorisation is based on the level of complexity that the model carries. Despite each of the above categories defining a respective involvement with the system undergoing

adaptation, there are a number of mutual characteristics that all systems exhibiting this behaviour have.

Having defined the types of models that can be created, as a next step the fundamental entities that will surface within each model need to be established. As defined by Holland [220], the process of undergoing adaptation requires the definition of a minimum of three fundamental entities for the modelling course to take place. These establish the centre for a formal framework, providing a fixed way of decomposing adaptive systems and are:

- The environment E undergoing adaptation
- The adaptive plan τ determining the changes within structures of the environment
- A measure of performance μ for the resulting structures in the environment

From the above, the environment represents the selection space for a solution to the problem under question. This does not imply that every point in this space needs to be premeditated beforehand but that every point in E can be uniquely identified in relation to others. Having defined this space, the adaptive system employing τ has as an objective to provide a selection mechanism within E for the mutation of the resulting structures. In order to achieve this, a measure of performance, μ , is necessary to distinguish between the performances of individual structures.

Pursuing further the logic of [220] to expand on the relationship between the environment E and the adaptive plan, the adaptive system is specified by the set of objects $\{\alpha, \Omega, I, \tau\}$, where:

- $\alpha = \{A_1, A_2, A_3, \dots\}$ represents the set of attainable structures within E
- $\Omega = \{\omega_1, \omega_2, \omega_3, \dots\}$ represents the set of operators for selecting structures in E
- I is the set of inputs to the system from the environment
- $\tau: I \times \alpha \rightarrow \Omega$ is the adaptive plan, which depending on I and α determines the operator to be applied to the system

Consequently, depending on the inputs, the system undergoing adaptation consists of a number of structures posing as solutions to the problem under question. Between these structures (illustrated in figure 4.5), a number of operators exist that allow for the selection of one structure through an adaptive plan and always depending on the measure of performance for each one.

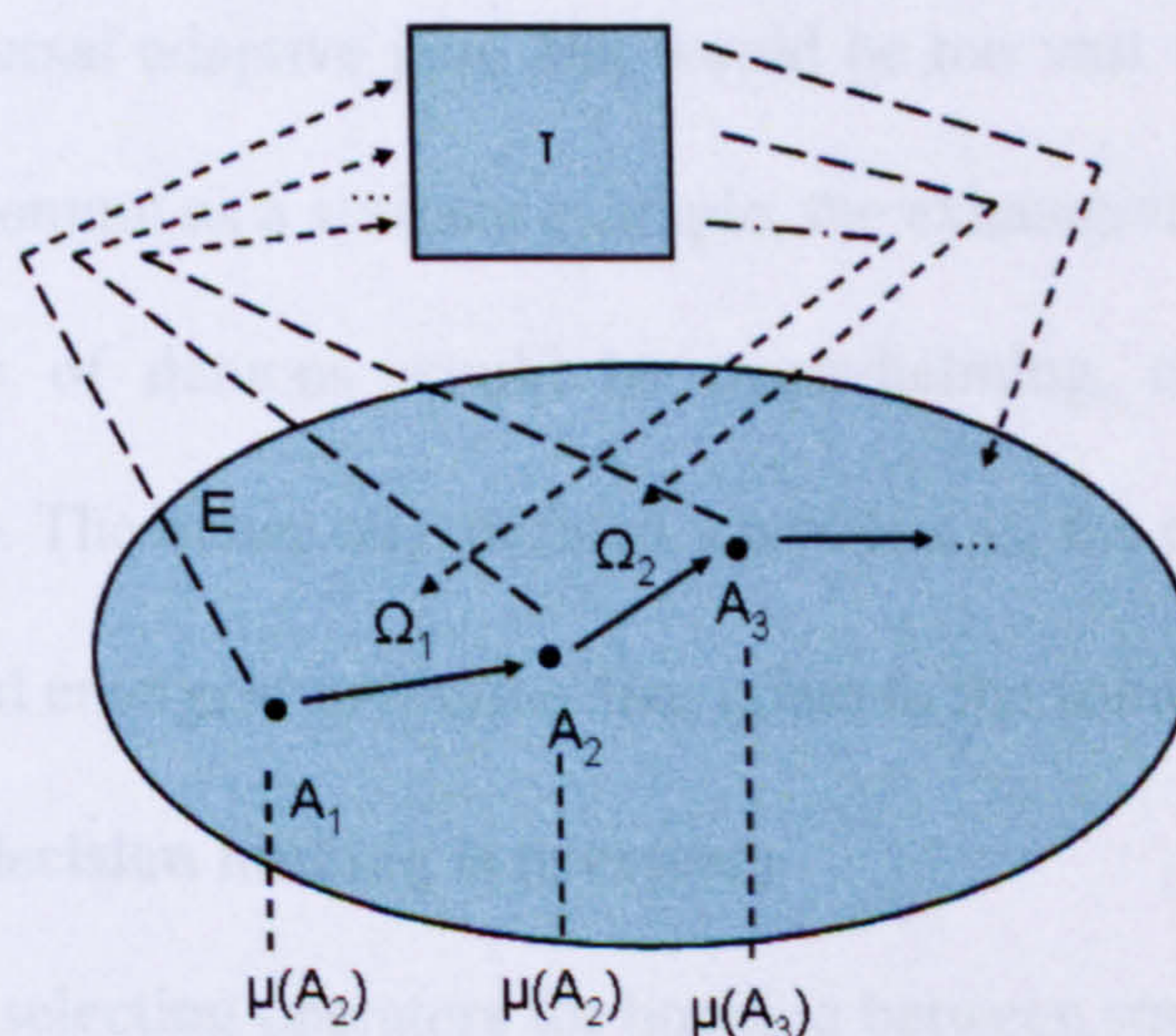


Figure 4.5: An abstract description of an adaptive system selecting iterative structures from an environment E , with an adaptive plan τ .

The generalised model above for an adaptive framework, presents a direct operational guide for the deployment of such a system. From this, provided that we question the limitations of the operation, as well as take into account any design pitfalls that might come into view, we have the ability to design a system that can host emergent behaviour. The remainder of this chapter focuses on factors relating to the operability of adaptive systems, both during the design, as well as the operational phases.

4.5 Limitations of operation

Deploying a system that has the ability to experience emergent behaviour, does not necessary imply that it will produce a viable and non-trivial solution to the problem at hand. For this, a set of requirements that guarantee functionality have to be put forward. From the development of related work, three fundamental requirements are presented below:

- **The problem should be well defined**

An ill defined problem can lead to too great a generalisation. This would result in a universal adaptive plan that would be too vast to manage. If we look at Pandemonium as a systems example, the exhaustive creation, testing and destruction of demons would be overwhelming, compared to any relative success. The better well defined a problem is, the easier an adaptive system can yield emergent properties that relate to the solution of it.

- **Unbiased decision making is necessary**

The process of selecting operators for hopping between structures within the environment E (figure 4.5) should be unbiased encouraging random

encounters [221] within the set of structures $\alpha = \{A_1, A_2, A_3, \dots\}$. Thus, the adaptive plan $\tau : I \times \alpha \rightarrow \Omega$ should carry a certain degree of uncertainty as a function for selecting structures from α . In terms of a Pandemonium, both the cognitive and processing demons of the respective layers two and three, as well as for the decision demon, should not make their decision solely based on the evidence that they have. For this, they might have a partial, as well as a completely negative view from the information that they have obtained. Furthermore, regardless of what partial states might yield, judging individual demons, as well as the entire Pandemonium should only take place after the necessary computations have been finalised, at every level within the system.

- **A currency should be introduced**

The measure of performance μ should be treated as a currency within the system, remaining unbiased from tampering from individual structures within E . Similarly, for a successful Pandemonium to operate correctly, a single tamperproof way of labelling the behaviour of demons should exist. If this were not the case, the four layer structure would fall apart, with demons claiming to have produced outputs that they would pass the decision demon, but yet fail to tackle the problem at hand.

As a conclusion, the above three points form an outline of the limitations that an adaptive system has. It needs to have a certain degree of freedom in the decision making process, enough to be able to iterate through potential solutions in an unbiased manner. At the same time it should also possess a well defined field of view. This leaves it capable of tackling the specified problem at hand and not

generalisations of it. Finally, an unbiased way of determining individual contributions at every level should exist. This, sharing many similarities with currency, should allow for the unprejudiced categorisation of demons within a Pandemonium.

4.6 Design pitfalls

Putting together the above limitations with the complexity that emergent systems can host, raises a number of pitfalls that should be avoided in the design process. This section documents these, circumventing the localisation to specific cases, in order to avoid loss of generality.

Firstly, as it is difficult to predict collective behaviour from individual rules, interrogating individual elements of the system will not yield in any useful information regarding the function of the group. The task of having a global view becomes inherently more difficult, as small changes in simple rules can totally alter the entire behaviour of the system. Consequently, a design pitfall to be avoided comes from establishing the correct feedback from the system and enabling the observer to have a global view of its behaviour.

This task is intuitively stated in the presence of a decision demon in every Pandemonium, as well as in the process of a well defined problem within the limitations of operation. Despite every layer in a Pandemonium having a number of demons, the top most layer (in an attempt to reside a level of control) hosts one demon responsible for making a decision within the system. Similarly, upon defining a problem, not only must the actual definition be taken into

account, but also ways of verifying the feedback of the system (not from individual entities) should be present.

Secondly, the division of labour (a concept surfacing from ants [15]) within the entities of the system should be inherent to the system. The trade-off that is established in this case presents the awareness of individual elements on the task at hand, versus the optimisation towards problem solving. Similarly to stronger ants carrying bigger loads, but not for the entirety of a journey, entities within an artificial system better at particular tasks should be given priority in performing them. Thus, the design pitfall of individual entities carrying the work load of the majority of the task that the system is attempting to perform should be avoided.

Finally, a rule that should always be noted is that simple commands are not always necessarily good. An example of this is the circular mill in army ants illustrated in figure 4.6. Despite the stigmergy present in the system with each ant following the pheromone trails in the environment, the entire colony appears to be stuck in an infinite circle of un-achievement¹².

¹² Beebe (1921) observed [222] a mill in Guyana that measured 1200 feet in circumference with a circuit time for each ant of about 2½ hours. The mill persisted for two days, with ever increasing numbers of dead bodies littering the route, but eventually a few workers straggled from the trail thus breaking the cycle, and the raid marched off into the forest.



Figure 4.6: A circular mill of army ants, stuck in an infinite loop circle, trailing on the pheromone patterns of other ants.

As a result, simple rules are required within the system, but their operation should be monitored in great detail to make certain that trivial or recursive states are not reached.

The dangers described within this section, even in their generality, form a boundary that further assists in a firmer definition of a system undergoing adaptation. Adding each one of these design pitfalls to an already well defined description allows us to deploy models that consent to the occurrence of emergence (through the surfacing of new properties) within the system.

4.7 Conclusions

The field of emergence does not carry a complete definition. The main reason for this fact is its relation to other areas that are still maturing within their domain. An example of this is the subject of complexity. Still, within SI,

emergent behaviour is an inherent characteristic, defining the observation of new properties in the problem solving process.

Baas offers a definition for emergence in terms of a system of observable structures, each carrying a number of different properties. In brief, a property that can be observed within a particular state, which (in the process of continuous change and adaptation) was not there before, is labelled as an emergent property for that particular structure.

Having given a definition for emergence, we are further motivated by the ability to design systems that can host such behaviour. For this, the design of Oliver Selfridge's Pandemonium is presented. Expanding on this, an introduction to adaptive system's modelling based on the work of Holland is given. This outlines the different characteristics that a system should have. The fundamental are: the environment E undergoing adaptation, the adaptive plan τ determining the changes within structures of the environment and a measure of performance μ for the resulting structures in the environment.

As deploying a system with the ability to experience emergent behaviour does not necessary imply that it will produce viable and non-trivial solutions, a number of limitations of operation are considered. Combining these limitations with the complexity that emergent systems can host gives rise to a number of pitfalls that should be avoided in the design process.

Hitherto, this chapter has presented in detail the field of emergence, how that can be defined and also how systems hosting such behaviour can be designed. Processes described here within, will be used as tools in the description of the system that was fabricated for the making of secure routing protocols for MANETs.

The design of the Swarm simulator system

5.1	Implementation decisions
5.2	Requirements specification: Problem description
5.2.1	Describing the MANET
5.2.2	Describing the adaptive pandemonium
5.2.3	Describing user interactions
5.3	Requirements analysis: Use case model
5.4	Static modelling of the problem domain
5.4.1	Modelling the network
5.4.2	Modelling the scenario
5.4.3	Modelling the protocol
5.5	Object structuring of sub-systems
5.6	Dynamic modelling
5.7	Swarm: An adaptive protocol selection system.
5.8	A Summary

This chapter presents the design methodology behind the software development of a discrete event simulation environment, capable of assessing wireless protocol designs. For this, the different sections of the system are identified and carefully described. Our modelling approach initially lays out the necessary characteristics for the static model and then proceeds into identifying the dynamic aspects that are required for the system to operate within the adaptive procedures identified in the previous chapter.

5.1 Implementation decisions

In proposing a wireless protocol for the security of routing in a MANET, the simulation of the protocol behaviour under one or more worst case scenarios offers a benchmark of performance for the resulting implementation. The protocol definition acts as an input to the simulator, which in turn provides a number of performance results, by offering a comparison for measures such as throughput, latency and so on. This process defines the functionality of a simulator in the development of a routing protocol stack. The first question that surfaces in the development of a new network simulator, relates to the reasoning for not using an existing event based simulator.

Although there is a number of simulation environments available for network research (e.g. QualNet, GloMoSim, SSFNet) the two dominant simulators for ad hoc networks are OPNET [223] and ns2 [224]. In OPNET, there exist three protocol definitions for MANETs, namely, DSR, AODV and ZRP, whilst ns2 after the completion of the MONARCH project [225], carries a more complete protocol package.

It is extremely difficult to expand the functionality of currently available event-driven, component based network simulators to facilitate adaptive behaviour. There are two main causes for this. Firstly, establishing a measure of performance for the protocol specification that can be fed-back as input to the system is a tedious process, dependant on a set protocol implementation. Even if we had the patience to extract the output data, redefining the protocol stack in real time would simply go beyond the capabilities of existing simulators. To

illustrate this, we choose the network simulator ns2. In this, the protocol stack is defined as a source file in C++, which is treated as a set usable entity by ns2. Altering this file automatically involves the recompilation of the source so that it can be used in the simulation process. Despite the advantages that ns2 has to offer, this process deprives us of the ability to dynamically alter a protocol implementation, thus limiting any operation within our adaptive environment.

Secondly, existing simulators require a full specification of every layer in order to simulate the communication process. This fact, as much as offering realistic results for existing standards, takes away from the generality of emergence whereby a set of simple rules generate the solution to a complex problem. Again, implementing this in ns2 would involve a full layer to layer specification which would grossly overcomplicate the problem definition within our environment.

Finally, similarly to network simulators, there exist simulation environments for emergent phenomena. Starlogo [226] is the prime example of such an environment, designed primarily to facilitate the construction of simulation models, rather than illustrative or analytical models. From the three categories of adaptive systems modelling (presented in section 4.4) simulation environments such as Starlogo focus primarily on minimal models for an idea, which can be later expanded [227]. Even though they can be applied to minimal models for a system, as well as system models, the required build-up in the design does not allow for a complete simulation of an entity as complex as a secure routing protocol within a MANET.

5.2 Requirements specification: Problem description

Following the decisions regarding the simulation environment, this section focuses on the system implementation. For this, the corresponding sections of the system are identified and a description of each one is provided. Further to this, the link with the remaining sections of the system is considered.

5.2.1 Describing the MANET

Commencing with the description of a MANET (section 1.3) we consider the network to operate in a two dimensional space. A network is ultimately a collection of communicating nodes. Each node, as an entity, has the ability to move unreservedly in the ad hoc networking environment present. The mobility of each node is determined by the corresponding mobility model [228 – 234] that it utilises. On the communication level, each node has a broadcasting radius of transmission, which is considered to be uniform and isotropic in time. A schematic of such a network description is illustrated in figure 5.1, below.

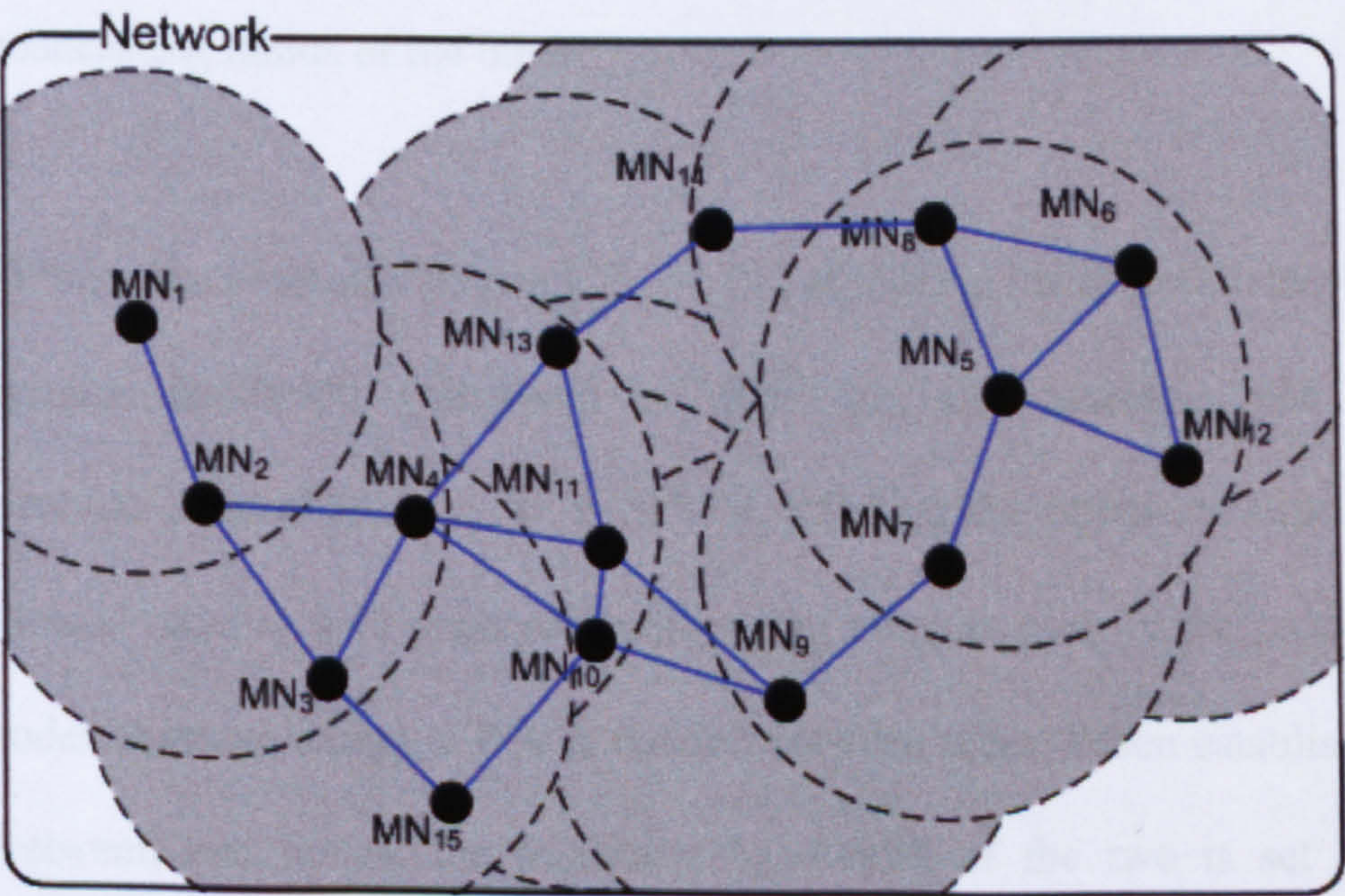


Figure 5.1: A two dimensional network schematic with 15 nodes, illustrating their corresponding links between them, as well as the broadcasting radius of each MN.

Nodes lying within the broadcasting radius of others yield the ad hoc network. A fundamental assumption is that every MN has knowledge of the existence of every other MN on the network, with each one being uniquely identifiable. Moreover, the communication actions of a mobile node are quite limited: Provided a link between other nodes exists, a MN can send, receive, drop or forward any packets that they receive. These actions come directly from the specification of ns2 [224].

As a typical scenario, a 50 node network topology as seen in [235] and [236] can be considered. Within this, and similarly to [94], a random waypoint mobility model [33] allowing pauses between the movement of individual MNs can be deployed. The maximum velocity for each node is generally in the range of $[0 - 20] \text{ ms}^{-1}$ for a space of $1500 \times 300 \text{ m}^2$ and assuming a bi-directional communication channel with a range of 200 m . From the assumption of the bi-directional channel, we say that a link between two nodes exists if one is within the broadcasting radius of the other.

For all links, the attributes seen in [23, 24, 55] of a delay (measured in ms) and a transmission bandwidth (measured in Mbs^{-1}) are also assumed. The delay represents an internal attribute for each MN, affecting the amount of time that a node would have to wait prior to commencing transmission. In the process of two nodes communicating, a link is created between them. When establishing a link between two nodes, the minimum bandwidth of the two is set as the bandwidth of the link. Prior to commencing each simulation, the bandwidth as well as the delay for each node will have to be selected at random. A range

often used offering realistic results in simulations is that of $[0.5 - 1.5] \text{ Mbs}^{-1}$ and $[0 - 500] \text{ ms}$ respectively.

In order to communicate, each MN deploys the available protocol specification for the given simulation. This, similarly to other network simulators, defines the steps that a node must follow in order to exchange information with any other node. Furthermore, the protocol specification also defines the node behaviour for each of three out of the four actions stated above; namely sending, receiving and forwarding packets. To handle packets, every node has an incoming and an outgoing buffer, ordered through a First-In-First-Out (FIFO) structure. This guarantees (in terms of processing) priority of older packets to newer packets with respect to time.

The action of packet dropping is considered as an attribute of malicious nodes, launching a passive attack on the network. Despite the fact that each node has the ability to become malicious, it can only act in a malicious manner if the specified scenario allows it to. Consequently, the two elements within the MANET section description that link to the adaptive section of the system are the ability for a MN to become malicious and the definition of the protocol specification being utilised during the simulation.

5.2.2 Describing the adaptive pandemonium

Following the requirement of an adaptive protocol design, the compartmental model description (illustrated in figure 1.2) is deployed. For this we identify the corresponding three layers of operation on which the system operates.

Starting from the bottom, layer 3 operates as the data processing layer, similarly to the data processing layer in Selfridge’s Pandemonium (section 4.3). Within this layer we distinguish the following three sub-systems, each with their corresponding attributes, illustrated in figure 5.2.

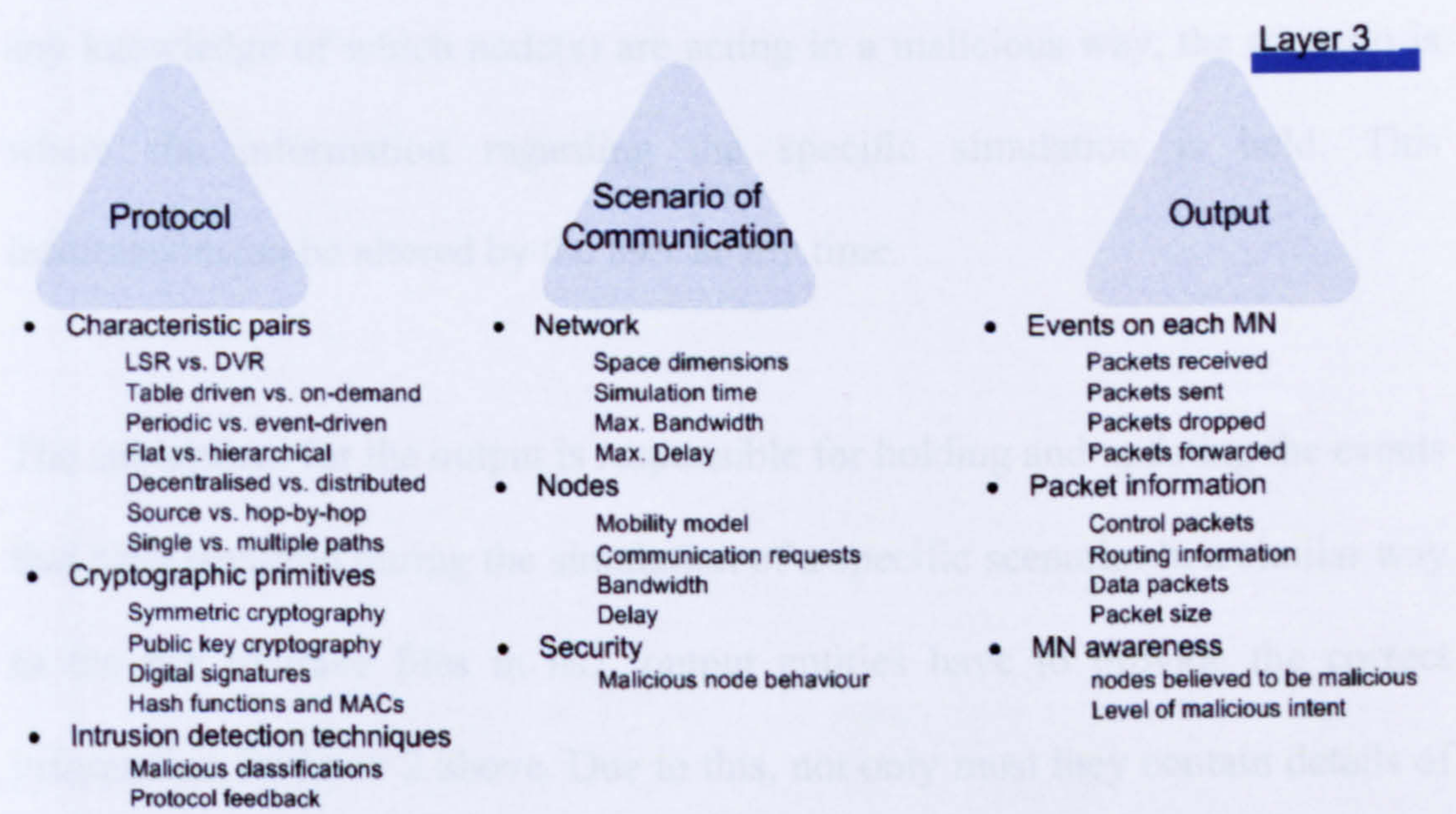


Figure 5.2: Layer 3 within the adaptive model, containing the sub-systems relating to the protocol description, the communication scenario and the output, each with their corresponding attributes.

The sub-system for the protocol generally holds the attributes seen in the classification of routing protocols (described in section 2.1). Extending on these, it also incorporates a number of security features focusing on the security mechanisms and defence techniques (described in section 3.5) available to each node. In particular, a number of cryptographic primitives are incorporated as features (described in sub-section 3.5.1) and also techniques for detecting the occurrence of an intrusion (described in sub-section 3.5.2) are embedded into the protocol description. We will examine each of these in greater detail in section 5.4, dealing with the static modelling of the problem domain.

The sub-system for the scenario deals with the communication scenario at hand. In general terms it has the ability of describing two or more nodes wanting to exchange information at a specific instance during the simulation and in the presence of nodes acting in a malicious manner. Despite of the nodes not having any knowledge of which node(s) are acting in a malicious way, the scenario is where the information regarding the specific simulation is held. This information can be altered by the user at any time.

The sub-system for the output is responsible for holding and updating the events that have occurred during the simulation of a specific scenario. In a similar way to the use of trace files in ns2, output entities have to provide the correct information for layer 2 above. Due to this, not only must they contain details of packets that transverse across the network but also information about the progress of individual MNs in realising the occurrence of malicious behaviour by means of the protocol deployed.

Layer 2 of the system describes the adaptive attributes that are responsible for the occurrence of emergent phenomena. Building on the entities identified in adaptive systems modelling (section 4.4) this layer incorporates two sub-systems, namely the adaptive plan and the measure of performance.

According to the description of adaptive models, further to the two above sub-sections there should also be a sub-section for the environment undergoing adaptation. The main reason why the environment is not elevated to a sub-section within this layer is as follows. In layer 3 one of the three sub-sections is

the protocol description; this covers every potential aspect of a full environment description remaining a layer below and thus attached to the communication scenario at hand. The spread of the fundamental entities of an adaptive system within the sub-sections of the system description in ordered layers can be seen in figure 5.3, below.

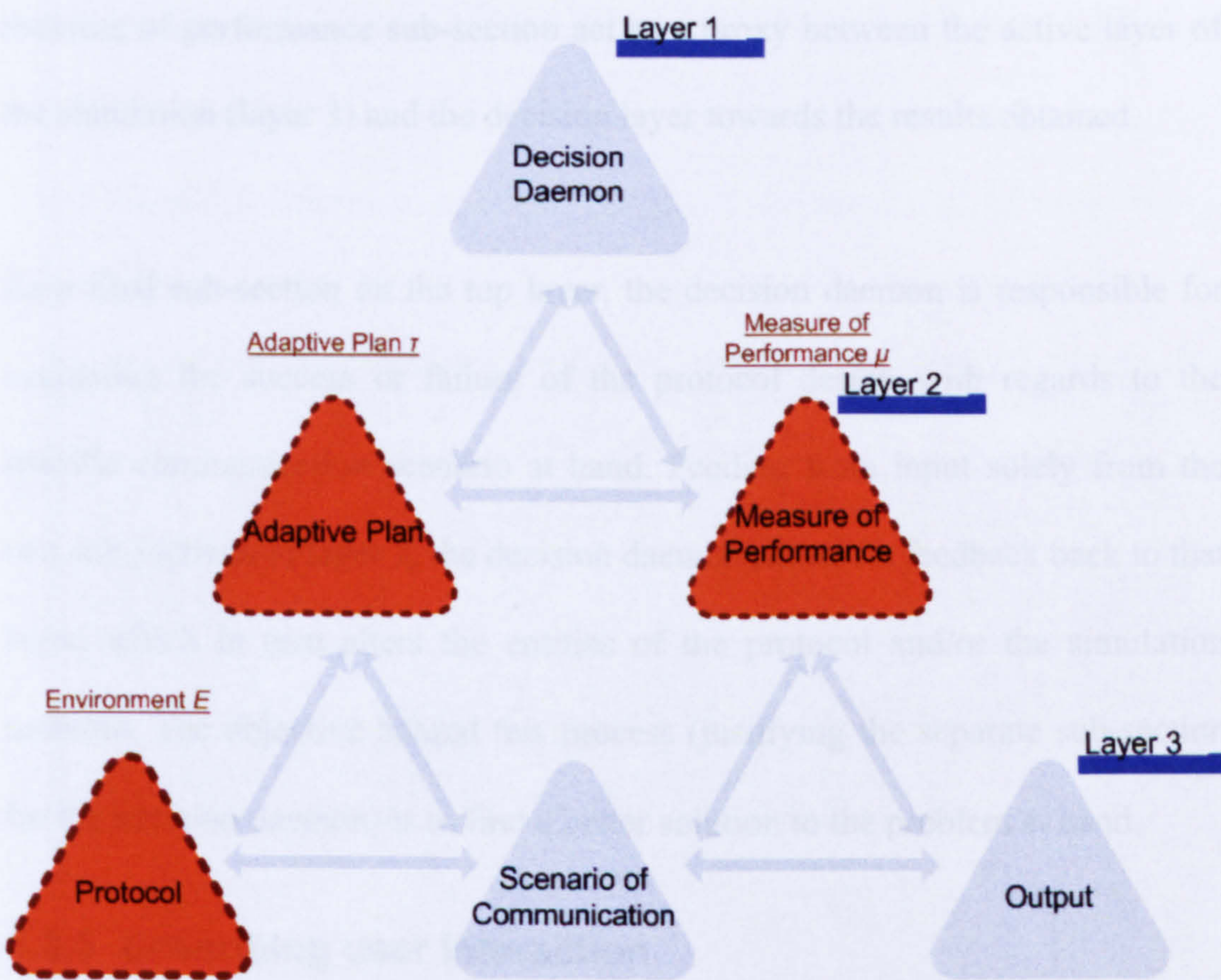


Figure 5.3: The labelling of the three fundamental entities, namely the environment, the adaptive plan and the measure of performance within the compartmental model description in its constituent layers.

Consequently, the adaptive plan feeds as a sub-system from information received by the protocol sub-system and the communication scenario undergoing simulation. Within the adaptive plan lies a description for identifying the different protocols that can be implemented yielding an abstract reference to the environment that is undergoing adaptation.

The sub-system for the measure of performance is responsible for providing the necessary feedback for emergence to take place. Unlike the environment which is described through the protocol and as the level of feedback that the system receives is vital to any decisions made regarding the protocol, a further sub-section is required to filter the output received. This filtering process makes the measure of performance sub-section act as a proxy between the active layer of the simulation (layer 3) and the decision layer towards the results obtained.

As a final sub-section on the top layer, the decision daemon is responsible for evaluating the success or failure of the protocol design with regards to the specific communication scenario at hand. Feeding from input solely from the two sub-sections of layer 2, the decision daemon passes its feedback back to that layer, which in turn alters the entities of the protocol and/or the simulation scenario. The objective behind this process (justifying the separate sub-section for the decision daemon) is to find a better solution to the problem at hand.

5.2.3 Describing user interaction

As much as this system operates independently in evaluating protocol designs, there is a necessary element of user interaction to indicate the settings regarding the initial communication scenario, as well as the levels of malicious intent. As a result, the user of the system should not only have the ability to evaluate the results obtained by the system but also to specify the scenario at hand and monitor its progress.

For this reason, the system is required to have a Graphical User Interface (GUI) that simulates specific network scenarios in discrete time intervals. Further to

this, a number of parameters involving the simulation, as well as the adaptive model being utilised would have to be able to be specified through the GUI. Unlike most simulators, this interface should not focus on the diagrammatic form of the network, yielding results on the events taking place, but on the results obtained from the process of adaptation upon protocol structures. Also, bearing in mind the way in which a protocol interacts with the system (figure 5.3), the process of how the information regarding the protocol is exchanged within the system should be clearly displayed. A proposed GUI for the system is depicted in figure 5.4 below.

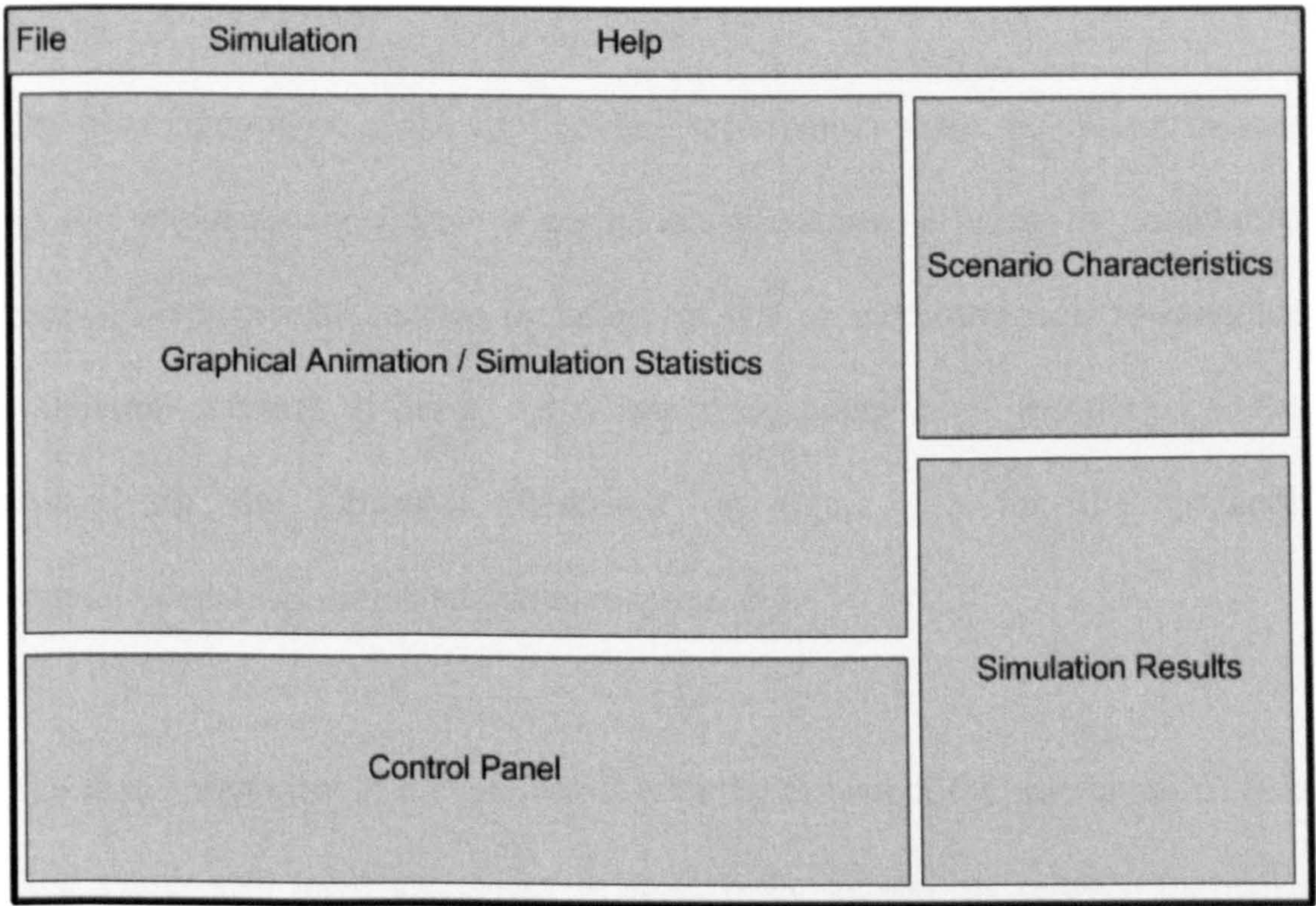


Figure 5.4: A schematic view of the GUI control and display panel, allowing the necessary user interactions.

Within the above figure, four distinct parts of the system are identified. Similarly to the network animator, nam, that operates within ns2 [224], the first component of the GUI is a pane on which the specific scenario, under which the protocol undergoing is being applied on, can be displayed. For this pane, we

consider two available views; a graphical animation view capable of illustrating the network topology in a similar way as that of figure 5.1 and also a simulation statistics view that can show at any one time instance the events taking place within the simulation.

The second component of the GUI that should be available for the user relates to the information of the communication scenario being deployed. This includes information about links in-between nodes, as well as all events due to occur at slotted time instances.

The third component of the GUI holding information about the system focuses on the output received from a particular simulation, offering the simulation results that show the success or failure of one or more protocols towards the simulation scenario at hand. All of the three components mentioned so far expand on the attributes illustrated on figure 5.2 for the protocol, communication scenario and output respectively.

The final component of the GUI deals with the control of the simulation. Within this, a user will have the ability to stop, reset, as well as re-trace particular events that have occurred in the component depicting the graphical animation of the system. This will also offer indirect control over the output and scenario component, as any action that might alter the simulation, will also affect the output and scenario of the system. To expand on this further, the following section looks at the model from an interacting user perspective, clarifying in greater detail the inputs and outputs that the system will have.

5.3 Requirements analysis: Use case model

Choosing to formally model the system in the Unified Modelling Language (UML) [237 - 239], we follow the modelling process described in the latest language specification [243]. UML is a family of graphical notations, backed by a single meta-model [241, 242] that assists in the design of Object-Oriented (OO) software systems. The specification of the language comes from the Open Management Group (OMG), which represents an open consortium of companies aiming to support and standardise the interoperability of OO systems. Currently, the latest specification release of the language is described in version 2.0 [243].

We commence our analysis by identifying the use case model of the system. Use case models [240], are a valuable tool to help comprehend the functional requirements of the system. Ultimately, their purpose is to clearly define the interaction that a user can have with the system implemented. Thus, the actions that the system is expected to perform, can be directly derived from the use cases, describing what exactly the system is attempting to accomplish.

The system description can provide the necessary information about the number of different uses that the designed system will have. From these, the number of individual actors, as well as their separate role can be identified. As seen in the previous section, there is no further necessity than a single actor of the system.

We label that actor as the *operator*.

Further to this, the operator through the GUI should have the ability to perform the following actions, which correspond to the *use cases* of the system.

- **CreateScenario:** Specify the communication scenario of the system
- **CreateProtocol:** Specify the protocol utilised within the scenario
- **FormatOutput:** Specify the output format for the system
- **ControlSimulation:** Control and monitor the progress of the simulation
- **QueryOutput:** Query the output received from the system

The system model derived from the above use cases is illustrated in figure 5.5.

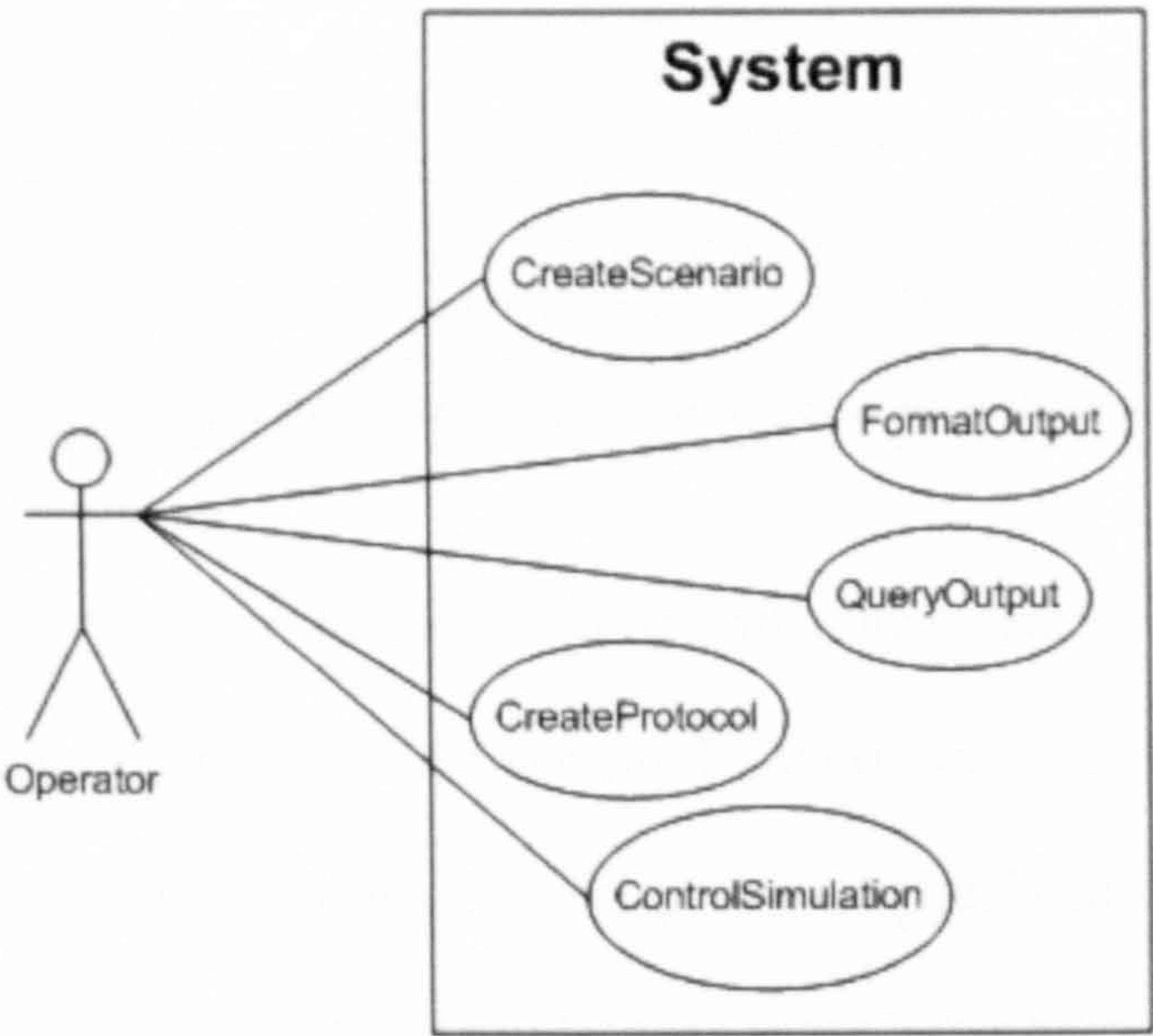


Figure 5.5: Adaptive Protocol Design System Use Case Model.

Following the order in which these are presented, tables 5.1 to 5.5 examine each of the above use cases in greater detail.

Table 5.1: Detailed description of the CreateScenario use case.

Field	Description
Use Case Name:	CreateScenario
Summary:	Create a valid communication scenario within a two-dimensional environment, involving a number of nodes that are attempting to communicate through the utilisation of a given protocol.
Actor:	Operator
Dependency:	CreateProtocol
Precondition:	System is idle, not undergoing any simulation

Description:	<ol style="list-style-type: none">1. <i>Specify the environment</i><ol style="list-style-type: none">a. <i>Specify the two dimensional space</i>b. <i>Specify the maximum broadcast radius</i>c. <i>Specify the node mobility model</i>d. <i>Specify the total simulation time</i>2. <i>Specify the network</i><ol style="list-style-type: none">a. <i>Specify the number of nodes</i>b. <i>Specify the maximum node delay</i>c. <i>Specify the maximum node bandwidth</i>3. <i>Specify the nodes</i><ol style="list-style-type: none">a. <i>Select the broadcast radius</i>b. <i>Select the delay</i>4. <i>Specify the scenario</i><ol style="list-style-type: none">a. <i>The malicious nodes</i>b. <i>The communication requests in time</i>c. <i>The existing links between nodes</i>
Alternatives:	<i>If one of the above four specifications is incomplete, the communication scenario fails to load within the system.</i>
Postcondition:	<i>A simulation is ready to begin within the system.</i>

This use case carries four fundamental specifications, each one of which is correlated to the performance of the system. Furthermore, it is dependant on the use case responsible for creating a valid protocol specification. The use case detailing the creation of a protocol can be seen in table 5.2 below.

Table 5.2: Detailed description of the CreateProtocol use case.

Field	<i>Description</i>
Use Case Name:	<i>CreateProtocol</i>
Summary:	<i>Specify the initial type of protocol that is to be used in the communication scenario. Within the specification, the range of characteristics which a protocol can obtain has to also be specified.</i>
Actor:	<i>Operator</i>
Dependency:	<i>None</i>
Precondition:	<i>System is idle, not undergoing any simulation</i>
Description:	<ol style="list-style-type: none">1. <i>Define the characteristic pairs</i><ol style="list-style-type: none">a. <i>LSR vs. DVR</i>b. <i>Table driven vs. on-demand</i>c. <i>Periodic vs. event-driven</i>d. <i>Flat vs. hierarchical</i>e. <i>Decentralised vs. Distributed</i>f. <i>Source vs. hop-by-hop</i>g. <i>Single vs. multiple paths</i>

	<ol style="list-style-type: none"> 2. Define which any cryptographic primitives <ol style="list-style-type: none"> a. Symmetric cryptography b. Public key cryptography c. Digital signatures d. Hash functions and MACs 3. Define the intrusion detection techniques <ol style="list-style-type: none"> a. Malicious classifications b. Protocol feedback
Alternatives:	<i>None. The characteristics define the broadest possible range that a protocol could have.</i>
Postcondition:	<i>A protocol is ready to be utilised by the scenario.</i>

From the three fundamental specifications that this use case has, partial characteristics can be specified within each one. This yields that a protocol could only comprise of certain characteristic pairs, without having to utilise cryptographic primitives or intrusion detection techniques in its specification.

Table 5.3: Detailed description of the FormatOutput use case.

Field	Description
Use Case Name:	<i>FormatOutput</i>
Summary:	Specify the format, as well as the amount of information in which any output generated from the system will be returned to the operator.
Actor:	<i>Operator</i>
Dependency:	<i>None</i>
Precondition:	<i>System is idle, not undergoing any simulation</i>
Description:	<ol style="list-style-type: none"> 1. Events on each MN <ol style="list-style-type: none"> a. Packets received b. Packets sent c. Packets dropped d. Packets forwarded 2. Packet information <ol style="list-style-type: none"> a. Control packets b. Routing packets c. Data packets d. Packet size 3. MN awareness <ol style="list-style-type: none"> a. Nodes believed to be malicious b. Level of malicious intent

Alternatives:	<i>None. The characteristics define the broadest possible range that the output format could have.</i>
Postcondition:	<i>The format in which a simulation will report is defined.</i>

Similarly to the CreateProtocol use case, an operator has the ability to define partial characteristics from the whole output format stated above. This use case carries three fundamental specifications, all of which are time dependant.

Table 5.4: Detailed description of the ControlSimulation use case.

Field	<i>Description</i>
Use Case Name:	<i>ControlSimulation</i>
Summary:	Have the ability to start, stop, pause, as well as query a particular simulation for results that it has produced.
Actor:	<i>Operator</i>
Dependency:	<i>QueryOutput</i>
Precondition:	<i>CreateProtocol, CreateScenario have completed</i>
Description:	<div>1. <i>Start, stop as well as pause the simulation</i> 2. <i>View a graphical animation</i> 3. <i>View simulation statistics</i> 4. <i>Query the simulation results</i></div>
Alternatives:	<i>None.</i>
Postcondition:	<i>The state of the system undergoing simulation can change accordingly, if the operator selects to start, stop, or pause the simulation.</i>

From this use case, the operator has complete control of the undergoing simulation, as well as the output that he can receive from it. In order to have the ability to control the system in its entirety, this use case utilises the QueryOutput use case, described in table 5.5, below.

Table 5.5: Detailed description of the QueryOutput use case.

Field	<i>Description</i>
Use Case Name:	<i>QueryOutput</i>

Summary:	Have the ability to query the system, for the output that it has generated in a particular simulation.
Actor:	<i>Operator</i>
Dependency:	<i>FormatOutput</i>
Precondition:	<i>The system is undergoing a simulation.</i>
Description:	<i>1. Query the system output relating to the simulation</i>
Alternatives:	<i>None.</i>
Postcondition:	<i>The output of the system is returned to the operator in the format that has been specified in the FormatOutput use case</i>

The above use cases clearly identify the actions of an operator within the system. Each use case has a number of fundamental specifications that outline the expected system behaviour. The section that follows details specific attributes of these actions focusing on the static modelling of the system. This comprises of the problem domain embracing the protocol descriptions, as well as the scenario of the simulation.

5.4 Static modelling of the problem domain

Following the conventional modelling approach to the static description of the model domain [239 – 241], we categorise the resulting classes of the OO analysis into two groups. The first group comprises of physical classes corresponding to physical structures within the system, while the second comprises of entity classes corresponding to entities that have to be fabricated as objects in order for the system to function cohesively. The joint set of the two groups together, details the static model for the system.

5.4.1 Modelling the network

The network corresponds to the group of physical classes that can model a MANET. In order to achieve this, we introduce the following classes:

- **Network class:** Is the group of nodes, holding information about the physical maxima, as well as the maxima relating to the exchange of information.
 - *Only one instance of the Network class exists:* Each simulation considers a single network
 - *One-to-many relationship with the Node class:* A network consists of a number of nodes
 - *One-to-many relationship with the Link class:* Within the network, a number of links between nodes can exist
 - *One-to-many relationship with the MobilityModel class:* A number of different mobility models can be utilised in a network
- **Node class:** Represents the individual MN attempting to communicate with other nodes which are part of the network
 - *Many-to-many relationship with the Link class:* A node can have a number of links
 - *One-to-one relationship with the MobilityModel class:* Every node has a single mobility model
- **Link class:** Represents the presence of a link between two nodes, once they are both within the broadcasting radius of each other
 - *Many-to-many relationship with the Node class:* A link is a connection between two nodes
- **MobilityModel class:** Describes the mobility model used by each node

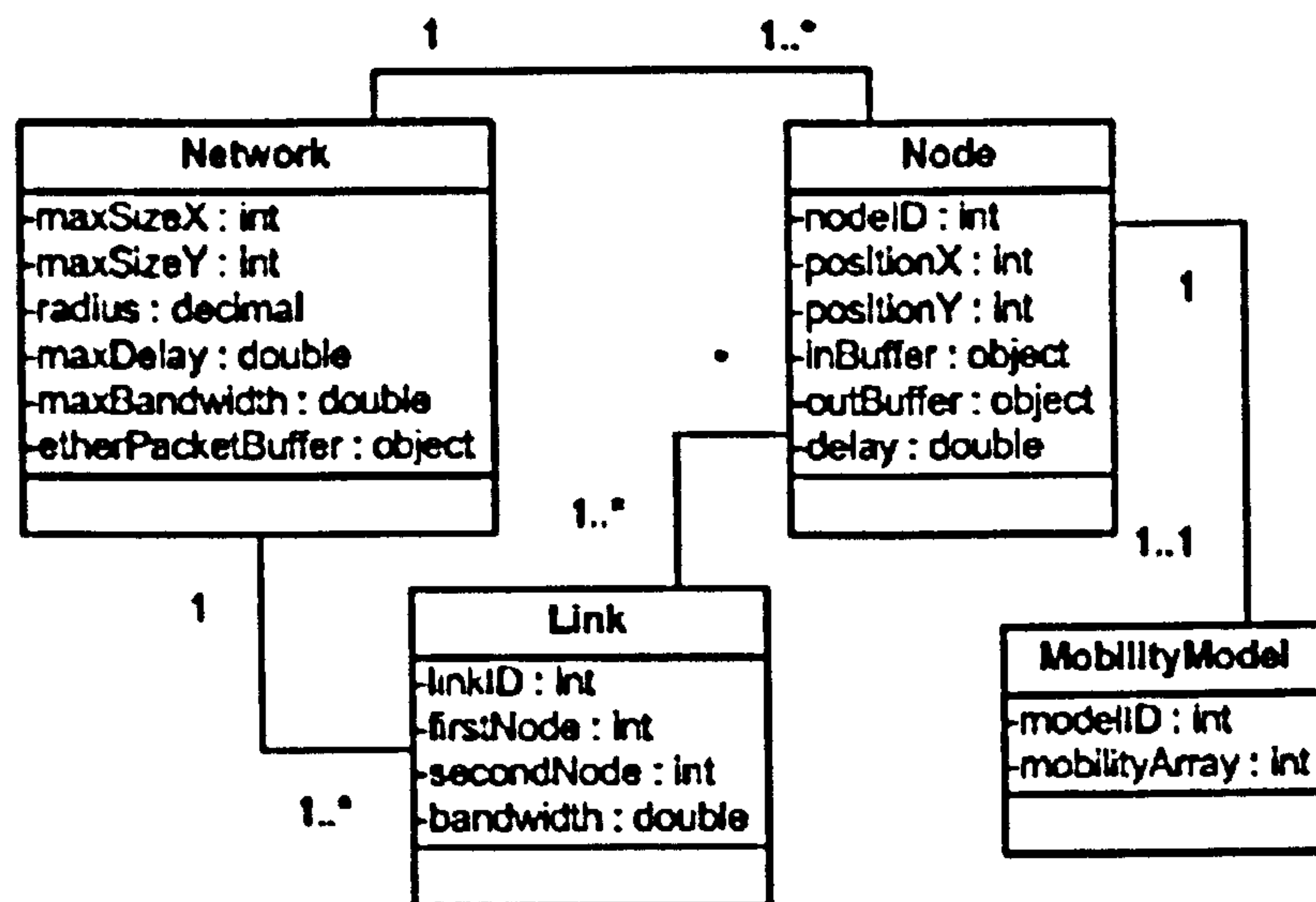


Figure 5.6: Conceptual static model for the network, corresponding to the physical classes that constitute a MANET.

For the above four classes, we identify the following class attributes, illustrated in figure 5.6 with the corresponding class relationships.

- Network: maxSizeX, maxSizeY, radius, maxDelay, maxBandwidth, etherPacketBuffer

A network has a two-dimensional region of operation (maxSizeX, maxSizeY) and also specifies the broadcasting radius of each MN. The maximum delay that each MN can have, as well as the maximum bandwidth that a link can experience are defined. Further to this, it holds a buffer on which the packets that need to be exchanged between nodes are temporarily stored; this buffer acts like an ether on the network.

- Node: nodeID, positionX, positionY, inBuffer, outBuffer, delay

A node is identified through a unique node ID on the network. At any one time instance every node holds a physical position within the network. Furthermore, every node experiences a transmission delay and also has an incoming and an outgoing buffer for receiving and transmitting packets.

- Link: linkID, firstNode, secondNode, bandwidth

A link is a communication link between two nodes, allowing them to exchange information. It carries a unique ID and also has a bandwidth that represents the rate in which the two nodes can send and receive data.

- MobilityModel: modelID, mobilityArray

Each mobility model represents the possible way in which a single or a group of MNs can change position within the network topology. Further to a unique ID for each model, an array of values is specified within each one, corresponding to particular characteristics in the movement.

5.4.2 Modelling the Scenario

The scenario corresponds to the group of entity classes that can recreate a communication scenario taking place within the network. In order to achieve this, we introduce the following four classes:

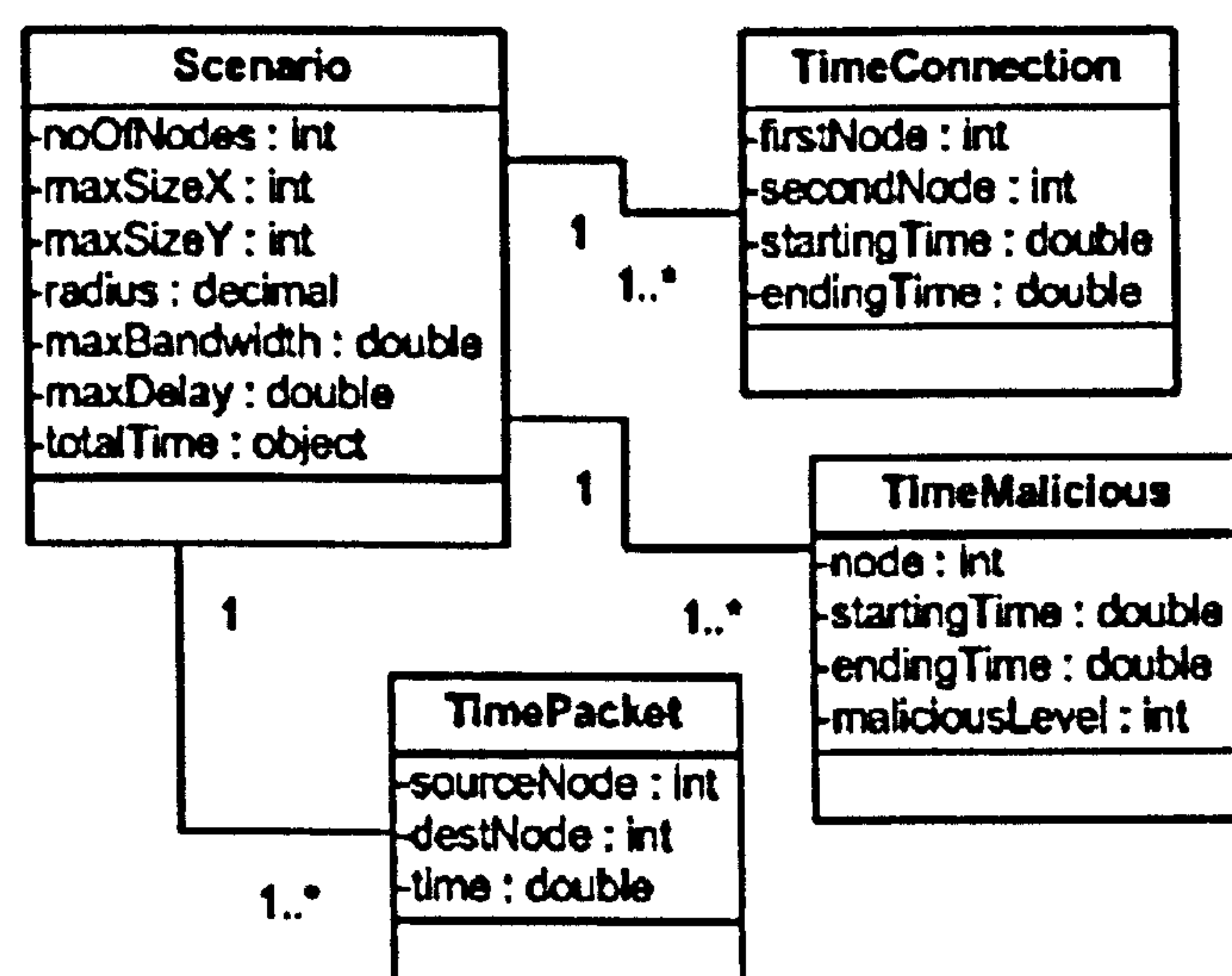


Figure 5.7: Conceptual static model for the scenario, corresponding to any communication scenario occurring within a MANET.

- **Scenario class:** This class groups together the necessary attributes that can describe a communication scenario taking place within a MANET. It is correlated to the time interval on which specific events occur on the network aiming to disrupt the communications between MNs.

- *Only one instance of the Scenario class exists:* Each simulation considers a single communication scenario
- *One-to-many relationship with the TimeConnection class:* A number of connections between nodes can occur during the simulation of a single scenario
- *One-to-many relationship with the TimeMalicious class:* A number of malicious nodes can exist during the simulation of a single scenario
- *One-to-many relationship with the TimePacket class:* A number of communication requests between two nodes on the network can occur during a single simulation.
- **TimeConnection class:** A TimeConnection represents the link between the two communicating nodes, including the start and end time of existence of this link. Time is represented in relative terms as a ratio between 0.0 and 1.0 of the total simulation time.
 - *It has to be stated that no relationship between this class and the TimeMalicious class, as well as the TimePacket class exists*
- **TimeMalicious class:** A TimeMalicious represents the time duration that a particular MN remains malicious and level of malicious intent that it has towards the network. Similarly to the TimeConnection class, time is represented as a ratio between 0.0 and 1.0 of the total simulation time.
 - *This class is independent of any TimeConnection instances or TimePacket instances that might exist in the scenario*

- **TimePacket class:** A TimePacket represents the request made from one node to another to communicate. Again, time is represented as a ratio between 0.0 and 1.0 of the total simulation time.

- *This class is independent of any TimeConnection instances or TimeMalicious instances that might exist in the scenario*

Similarly to the previous sub-section, we identify the following class attributes, illustrated in figure 5.7 with the corresponding class relationships.

- **Scenario:** noOfNodes, maxSizeX, maxSizeY, radius, totalTime, maxDelay, maxBandwidth

As a scenario embraces all the required characteristics for the exchange of information to take place, it carries all the attributes of the Network class, as well as the total time for which a simulation is going to occur (totalTime) and the total number of nodes (noOfNodes) that constitute the network. In a way, the scenario class is the back end to the front end of the network class.

- **TimeConnection:** firstNode, secondNode, startingTime, endingTime

Corresponding to the physical links that exist within the network, a TimeConnection details which nodes form the link and also the starting and end time for which the link exists. Note that time is considered as a ratio of the total simulation time always being within the range of 0.0 and 1.0.

- **TimeMalicious:** node, startingTime, endingTime, maliciousLevel

A MN (node) can act in a malicious manner, not obeying the rules set out by the protocol for exchanging information in the network. The duration for which the node remains malicious is specified through a starting and an end time. Based on the descriptions of section 3.3 classifying a malicious attack,

each TimeMalicious instance has a level of adversarial intent, set by the maliciousLevel attribute. Table 5.6 details the values that this variable can obtain with respect to the malicious ability experienced within the network.

- TimePacket: sourceNode, destNode, time

Each node can attempt to communicate with any other node that is part of the network at any instance during the simulation. The TimePacket class specifies this, having as attributes the source MN (sourceNode), the destination MN (destNode) and the time that this takes place. Once again, time is considered in relative terms, as a ratio between 0.0 and 1.0.

Table 5.6: The definition of the malicious level attribute, as a five digit number for every TimeMalicious class.

	Active - n -m attacks	Adaptivity and intelligence of an adversary	Collusion of power		Malignancy of attacks
Malicious Level	Digit 1	Digit 2	Digit 3	Digit 4	Digit 5
			Number of colluding nodes	Level of cooperation	
0	Active – 0 – 1	Completely static adversary	None	None	No adversarial behaviour
3	Active – 0 – x	Oblivious dynamic adversary	Up to 30% of the MNs	Up to 30% of the information	Benign adversary
6	Active – 1 – x	Adaptive adversary	Up to 60% of the MNs	Up to 60% of the information	Static malicious adversary
9	Active – y – x	Adaptive selective adversary	Up to 90% of the MNs	Complete share	Proactive malicious adversary

Within the above table, further clarification is needed to understand how the two dimensions relating to collusion of power have been categorised. For the number of colluding nodes, we consider the percentage of the total nodes within

the network that are participating in a malicious collusion. Similarly to this classification, for the level of cooperation in-between MNs, we consider the amount of information (in percentage of the total) that each node discloses to other nodes within the MANET. Furthermore, we consider this scale to be linear; as a result, a value of 5, on digit 4 of the *maliciousLevel*, would imply that the specific node would only disclose half of the known information that it considers valid, to other nodes. Specific content would include hop-count, routing tables, hash values, etc.

Even though the characteristic of a *maliciousLevel* is an individual attribute for each node that is part of the network, some digits of this number have to remain identical for every node, as they represent group behaviour in the maliciousness of an attack. These are digit 1 and digit 3, representing the active – n – m attacker hierarchy and the number of colluding nodes within the network. As an example of how the value of the *MaliciousLevel* relates to the behaviour of an individual MN, consider that, say, *MaliciousLevel* = 00003. This would imply that we are modelling a scenario involving an active – 0 – 1 attack of a completely static adversary, colluding with no other nodes and classified as a benign adversary; in brief, the MN is a passive eavesdropper on the network.

5.4.3 Modelling the protocol

The protocol corresponds to the physical class that represents the wide range of operation that a protocol of communication for a MANET can carry. For this, we proceed into defining how the characteristics of section 2.1, as well as those relating to intrusion detection and cryptographic primitives identified in subsection 5.2.2 (and grouped in figure 5.2) can be fitted into a class description.

In order to be able to safeguard from a potential attack, the behaviour of the attacker must be noticeably different from that of a normally behaving router [244]. This forms the basis for the deployment of Intrusion Detection Systems (IDSs) [245, 246], which aim to detect traffic anomalies, as well as misuse patterns within a networking environment. Detecting misbehaving routers has also been studied on the protocol level, in terms of routing as seen in [247, 248]. Even though malicious behaviour such as the black hole problem facilitates only a single, passive state from an attacker's perspective, offering only the choice of dropping packets, there is still a number of detection states that can act as a measure of performance with respect to the routing protocol under question. Thus, with the occurrence of a malicious event, neighbouring MNs can:

- *Not realise the occurrence of the malicious event*

Assume the exchange of information has taken place successfully and in cases continue to send packets along the same route. This is by far the worst case scenario, where the routing protocol fails in all aspects to deal with the occurrence of a malicious event.

- *Realise a malicious event is occurring within the network*

Understand that data traffic is being manipulated somewhere along the route without having the ability to narrow down which node is behaving in a malicious manner.

- *Realise which MN(s) is causing the malicious event*

Not only have the ability to realise the occurrence of a malicious event, but also be able to identify which MN(s) is causing it.

- *Avoid information originating from a known malicious node*

Having realised that a malicious event is present along a specified route, perform any necessary action that allows for the re-routing of information ignoring incoming traffic from particular malicious nodes.

- *Isolate information originating from any malicious node*

Having realised which MN(s) is performing the attack and exploited different paths into rerouting data, attempt to move to an offensive state towards specific MN(s) with the objective of isolating them through typically advertising its existence on the network.

The above five cases act as a measure of performance for the occurrence of adversarial behaviour within the MANET. In an attempt to take into consideration redundant routing information (thus simplifying any detection taking place), we assume that there exists a valid solution to achieve any of the above states and thus at least one alternative route to the destination exists within the resulting topology at any moment in time.

Having presented the characteristics that define a wireless routing protocol (section 2.1), a way of representing individual criteria, as well as hybrid protocol designs for each pair is required. This would provide the necessary environment undergoing adaptation where the resulting protocol structures would attempt to tackle the malicious scenario at hand.

Taking a probabilistic approach into this classification, we define the level l for each characteristic pair, as a real value within the range of $[0, 10]$. This dimensionless number yields the bias of the protocol between the two criteria of each characteristic. We consider the level l , to be dependant on a partial attribute h , measured in proportion to a total attribute n , and the probability p , of a particular event to occur. Thus, the level l is defined in (5.1) as follows:

$$l = \left\lfloor \frac{10 \times h}{n} \right\rfloor + p \quad (5.1)$$

Where:

$l \in \{x \in \mathbb{R}, 0 \leq x \leq 10\}$ represents the level for the characteristic pair.

$h \in \{x \in \mathbb{R}^+, x < n\}$ is the partial attribute representing a value directly related to the network.

$n \in \{x \in \mathbb{R}^+\}$ is the total attribute in the dimensions of h .

p is the probability of a given event regarding a specific state of either attribute h or n . Note that for $l=0$ and $l=10$, we also assume that $p=0$.

By inspection, we can see that each characteristic is related to, time (periodic vs. event driven updates), the number of nodes participating in a particular action (LSR vs. DVR) or the number of routes present (single path vs. multiple paths). Consequently, we proceed into defining each of the above attributes h and n as well as the probability p for every pair.

- LSR ($l = 0.0$) vs. DVR ($l = 10.0$)

The level l within this characteristic represents the distance in hop counts, within which an originating MN will take into consideration (for routing) the reply information received. Also, within the partial distance vector received, information only up to l hops away will be taken into account. Thus, the attribute h represents the hop distance for which information will not be neglected. On the other hand, the attribute n is the maximum hop distance excluding any routing loops present on the network. The probability p in this case is the probability that a MN will take into consideration (either in the distance vector or in the RREQ packet) information which is $(h + l)$ nodes away.

- On-demand routing ($l = 0.0$) vs. table-driven routing ($l = 10.0$)

The level l within this characteristic represents the distance in hop counts for which a routing table will be held and thus a route request will not be launched. Hence, the attribute h represents the hop distance for which a routing table will be stored, while the attribute n , again, is the maximum hop distance, excluding any routing loops present on the network. The probability p in this case is the probability that a MN will store (and not request) information which is $(h + l)$ nodes away.

- Event-driven update ($l = 0.0$) vs. periodical update ($l = 10.0$)

The level l within this characteristic represents the proportion of updates that are received in slotted time intervals, compared to the ones which are received in the occurrence of an event. As time is the contributing factor of this characteristic, the attribute h represents the number of updates (per unit

time) exchanged on the network on a periodic basis, while the attribute n represents the total number of updates (per unit time) received by each node. The probability p in this case is the probability of accepting $(h + l)$ periodic updates in unit time.

- Flat structure ($l = 0.0$) vs. hierarchical structure ($l = 10.0$)

The level l within this characteristic represents the ideal number of partitions relative to the total number of nodes, which the protocol would (depending on the available links) separate the network, for optimum protocol performance. Hence, the attribute h represents the number of nodes that belong to the various clusters, with the attribute n representing the total number of nodes on the network. The probability p in this case represents the probability of attaching a single extra “flat node” to an existing cluster group during the total time of the simulation.

- Decentralised computation ($l = 0.0$) vs. distributed computation ($l = 10.0$)

The level l within this characteristic represents the average number (compared to the whole) number of nodes that engage in the distributed computation of a route across the network. Thus, the attribute h represents the number of nodes that will actively engage with others to help them define a route to a specified node. The attribute n represents the total number of nodes on the network. In this case, the probability p represents the probability of a single extra MN to request the assistance of others during the total time of the simulation.

- Hop-by-hop routing ($l = 0.0$) vs. source routing ($l = 10.0$)

Similarly to LSR vs. DVR, the level l within this characteristic represents the number of slots available within a packet that could be occupied with the route followed. This list would have to be ordered, with either a FILO or a First-In-First-Out (FIFO) structure. In this case, the attribute h represents the number of additional slots available within the packet, while the attribute n represents the maximum number of slots that could be made available through a specific route. Once more, the probability p represents the probability of adding a single extra slot within each packet during the total time of the simulation.

- Single path ($l = 0.0$) vs. multiple paths ($l = 10.0$)

Finally, the level l within this characteristic represents the number of routes that will be made available for a single request made on the network. In this case, the attribute h represents the number of extra routes that the originating MN could take into account, with the attribute n is representing the total number of available routes at that moment in time. Once again, the probability p in this case is the probability of considering a single extra route for a particular route request.

To understand the meaning of this range, in the case of table driven routing versus on-demand routing, the value of $l = 0.0$ would represent a protocol which is solely on demand initiated, while a value of $l = 10.0$, one that is exclusively table driven. A value of $l = 3.7$ would give rise to a hybrid protocol that despite of holding a routing table for some nodes, would initiate a route request procedure for the vast majority of routes needed on the network.

The decision for the range of the level l is influenced by three contributing factors. Firstly, in most cases, the attributes h and n relate to the total number of nodes, present on the network. In order to be able to describe resulting level values in a meaningful way we would favour the attribute n to always be greater than the maximum level value. Even though this is not a requirement and despite the use of just 6 nodes in [6] to describe a black hole scenario, standard Ad Hoc routing scenarios for MANETs [77], seen in [235] and [236] typically involve a larger number of MNs.

Secondly as a probability measure is present to signify small decision making, there is a definite need for a measure that is one order of magnitude greater than that. A last contributing factor relates to computation and the fact that all values (to one decimal place) within the specified range can be represented well within half the size of the smallest primitive data type in Java [249], that of a byte.

As we shall see in chapter 6, depending on the results obtained in each simulation scenario, the range of each level can be quantised accordingly. We consider each characteristic independent of one another; knowing the value for the table driven versus on demand initiated does not yield any information with respect to the protocol being, say, a single path versus a multiple path design.

From the above, we define the protocol class having the following attributes. Furthermore, giving the protocol class the ability to secure each protocol structure, for each cryptographic primitive we define a corresponding class with

the respective attributes. Hence, the protocol class can utilise each of the cryptographic classes defined, provided it takes on the burden of defining and appropriately disclosing specific attributes.

- **Protocol class:** Defines the protocol behaviour through the above stated numeric attributes involving intrusion detection states, as well as all the above described protocol characteristic pairs. Each protocol can use a number of the following cryptographic instances in its definition.
 - *Has a one-to-many relationship with the Sym_cipher, Asm_cipher, Dig_signature and Hash_mac class:* Each protocol can utilise these primitives for control data on the network
- **Sym_cipher class:** Represents the ability to encrypt and decrypt data symmetrically that are part of the communication protocol, through a secret and shared key.
- **Asm_cipher class:** Represents the ability to encrypt and decrypt data in an asymmetric way, through a public and private key pair.
- **Dig_signature class:** Represents the ability to digitally sign data, as well as verify data that has being signed, using a private and public key pair.
- **Hash_mac class:** Represents the ability to create as well as authenticate a message digest or MAC on control data being exchanged.

Similarly to the previous sub-section, we identify the following class attributes, illustrated in figure 5.8 with the corresponding class relationships.

- **Protocol:** intrusionDetectionLevel, linkStateVsDistanceVector, tableVsOnDemand, periodicVsEventDriven, flatVsHierarchical, decentralisedVsDistributed, sourceVsHopbyHop, singleVsMultiplePaths

A protocol is categorised in terms of each of the attributes specified in the class, corresponding to the seven characteristic pairs as a decimal number between the range [0.0, 10.0], as well as the intrusion detection level, which corresponds to an integer between [1, 5]. Further to this, the protocol class can utilise a number of instances of any of the cryptographic primitive classes that follow.

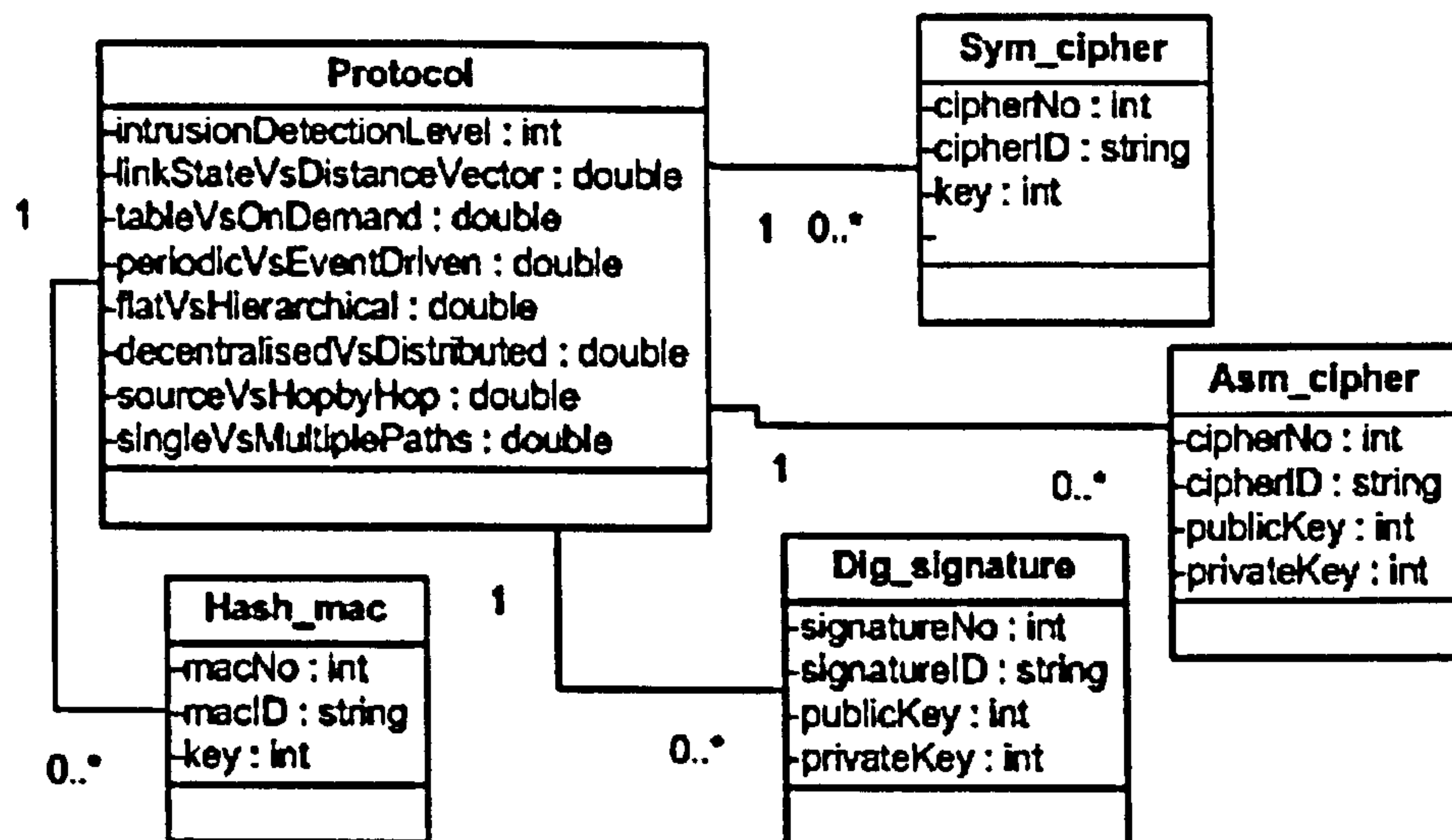


Figure 5.8: Conceptual static model for the protocol, illustrating the use of the corresponding cryptographic classes.

- Sym_cipher: cipherNo, cipherID, key

The symmetric cipher class (sub-section 3.5.1.2) has a unique number (cipherNo) corresponding to the particular instance, a cipher ID corresponding to the block cipher that it uses, as well as a key used for the encryption and decryption process.

- Asm_cipher: cipherNo, cipherID, publicKey, privateKey

The asymmetric cipher class (sub-section 3.5.1.3) has a unique number (cipherNo) corresponding to the particular instance, a cipher ID corresponding to the asymmetric cipher that it uses, as well as a public and a

private key pair, used for the encryption and decryption process respectively.

- **Dig_signature:** signatureNo, signatureID, publicKey, privateKey

The digital signature class (sub-section 3.5.1.4) has a unique number (signatureNo) corresponding to the particular instance, a signature ID corresponding to the signing scheme used, as well as a private and a public key pair, used for the creation and authentication of the signed message.

- **Hash_mac:** macNo, macID, key

The class for the creation of hashes and authentication codes (sub-section 3.5.1.5) has a unique number (macNo) corresponding to the particular instance, a MAC ID corresponding to the particular scheme used, as well as a key used for MACs and otherwise set to a default value.

5.5 Object structuring of sub-systems

Having presented the static model for the system and identified the corresponding physical, as well as entity classes, a way of ordering them within the system is required. This section presents the sub-systems identified and also yields the structure of each one in terms of their containing objects. For this, an overview of the classes seen in the previous section together with the objects required for the dynamic modelling of the system are grouped together giving a systems' overview.

We begin by labelling the entirety of the system as Swarm, stemming from SI. This system is required to have a GUI, showing the topological movement of nodes under a specific mobility model in a communication scenario. As the unique element within Swarm is the adaptive side of the simulation, the system

will have to embrace the layered approach illustrated in figure 5.3 and within this, incorporate remaining physical classes relating to the dimensions of our protocol analysis. Grouping relevant classes into packages, the package dependencies of the system can be seen in figure 5.9 below.

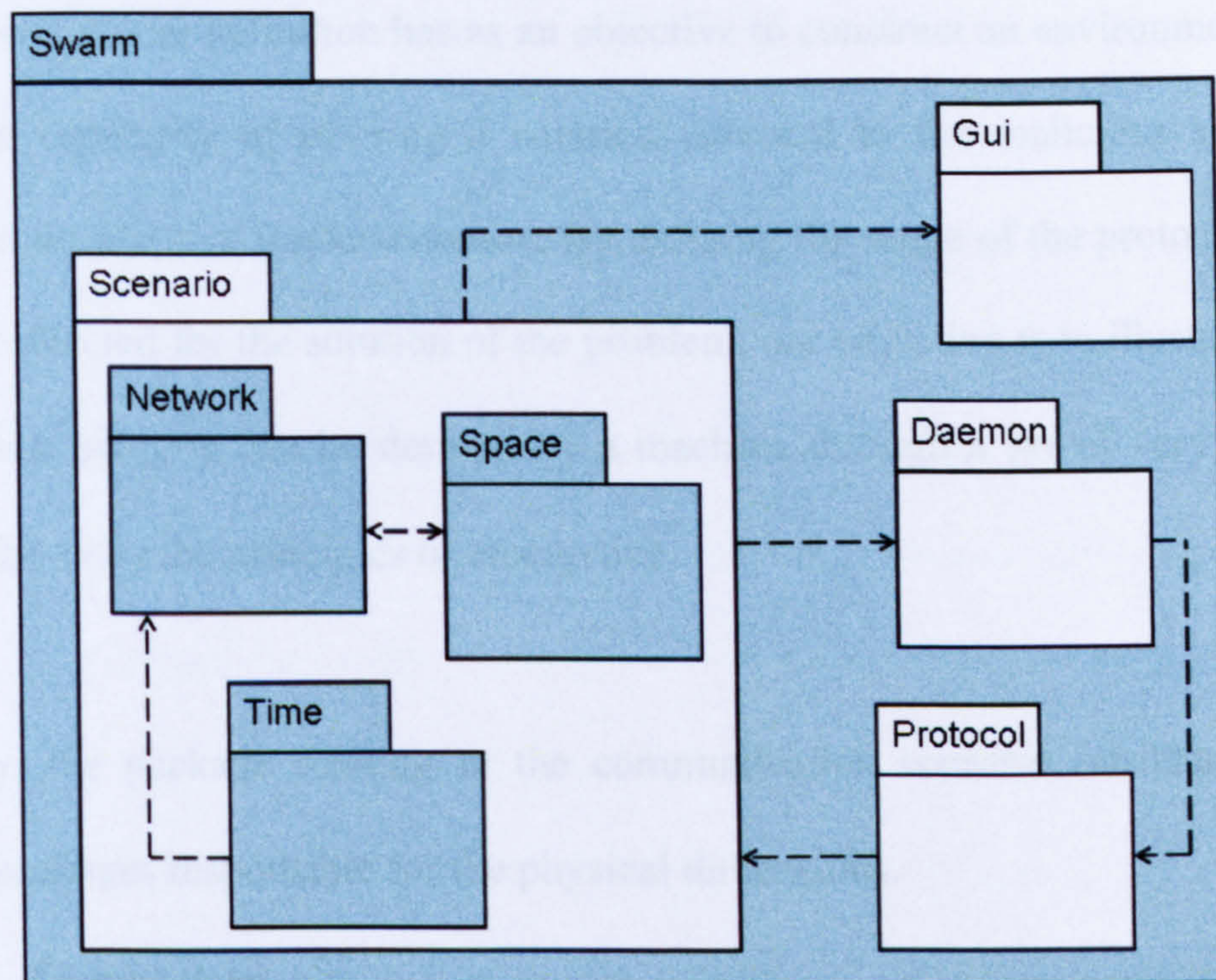


Figure 5.9: A system overview in UML of the package dependencies within Swarm.

Swarm contains a core network component which is utilised by the simulation scenario. This scenario carries the necessary descriptions of the communication phases that represent the occurrence of one or more malicious events on the network. Further to this, the system reports to the daemon package, representing the layer 1 of figure 5.3. In turn this layer propagates the information to the protocol specification (through the layers of the model) which is then modified accordingly for the scenario. Once a protocol has been selected from the initial environment and attached to a specific scenario, a decision daemon is responsible for applying the adaptive plan, depending on the success or failure

of the protocol in question. The entire process of the communication scenario deployed, as well as the protocol used within it is reported back to the GUI for the user.

The above model definition has as an objective to construct an environment that has the capability of offering a solution protocol to the malicious scenario through an adaptive implementation. By defining the range of the protocols that can be selected for the solution of the problem, our objective is to illustrate that the fittest solution can be derived by a machine through a set of very simple rules that obey the principles of emergence.

Finally, the package relating to the communication scenario introduces two more packages responsible for the physical dimensions.

- *Time package*
 - **Time class:** Defines how time is measured within a communication scenario. The standard time format used is *hh:mm:ss:csc*, where *hh* is the number of hours in the range [00 – 99]; *mm* is the number of minutes in the range [00 – 59]; *ss* is the number of seconds in the range [00 – 59]; *csc* is the number of cent-seconds in the range [000 – 999].
 - **Clock class:** Defines a clock that can measure Time in the above specified way, within a given range. The range of a Clock identical to that of Time [00:00:00:000 – 99:59:59:999].

- *Space package*
 - **Position class:** Defines the entity of position in two dimensions within which a Node is present, corresponding to a pair of positive Cartesian coordinates.
 - **Space class:** Defines the region (in terms of a maximum Position) within which all nodes that are part of a network can change Position in.

Through the definition of the above two packages, the MNs of the network in which the scenario is deployed, obtain the ability of movement in time. This places the emphasis of communication to the unique attributes of a MANET, in which the communication links between nodes vary dynamically depending on their position. Thence, we proceed to the modelling description of the system as that operates dynamically in real-time, attempting to simulate and categorise the performance of protocols within the communication scenario.

5.6 Dynamic modelling

The dynamic modelling of the system in UML entails determining the sequence of events in the complex use cases within the system [238, 239]. For this, the use of sequence diagrams is required. Sequence diagrams enable us to view the behaviour (in real-time) of several objects and their interactions within the system. Thus, in this section we present the sequence of object interactions in the creation of a communication scenario as well as the way in which a protocol is assessed once a simulation has begun. The former, expands on the interaction of the static entities that encompass the simulation, while the latter presents the way in which adaptation is applied to the selection process.

We begin by reviewing the sequence of events for the creation of a communication scenario. As illustrated in figure 5.10, the operator selects the creation of a new scenario from the Gui. This triggers the creation of a Space and a Clock object, with the Space requiring the specification of the maximum 2-D coordinates and the broadcast radius and the Clock requiring the specification of the maximum time. Once this process is complete the operator has to provide the specification of the communication scenario. This involves the number of nodes, as well as the number of malicious nodes present on the network.

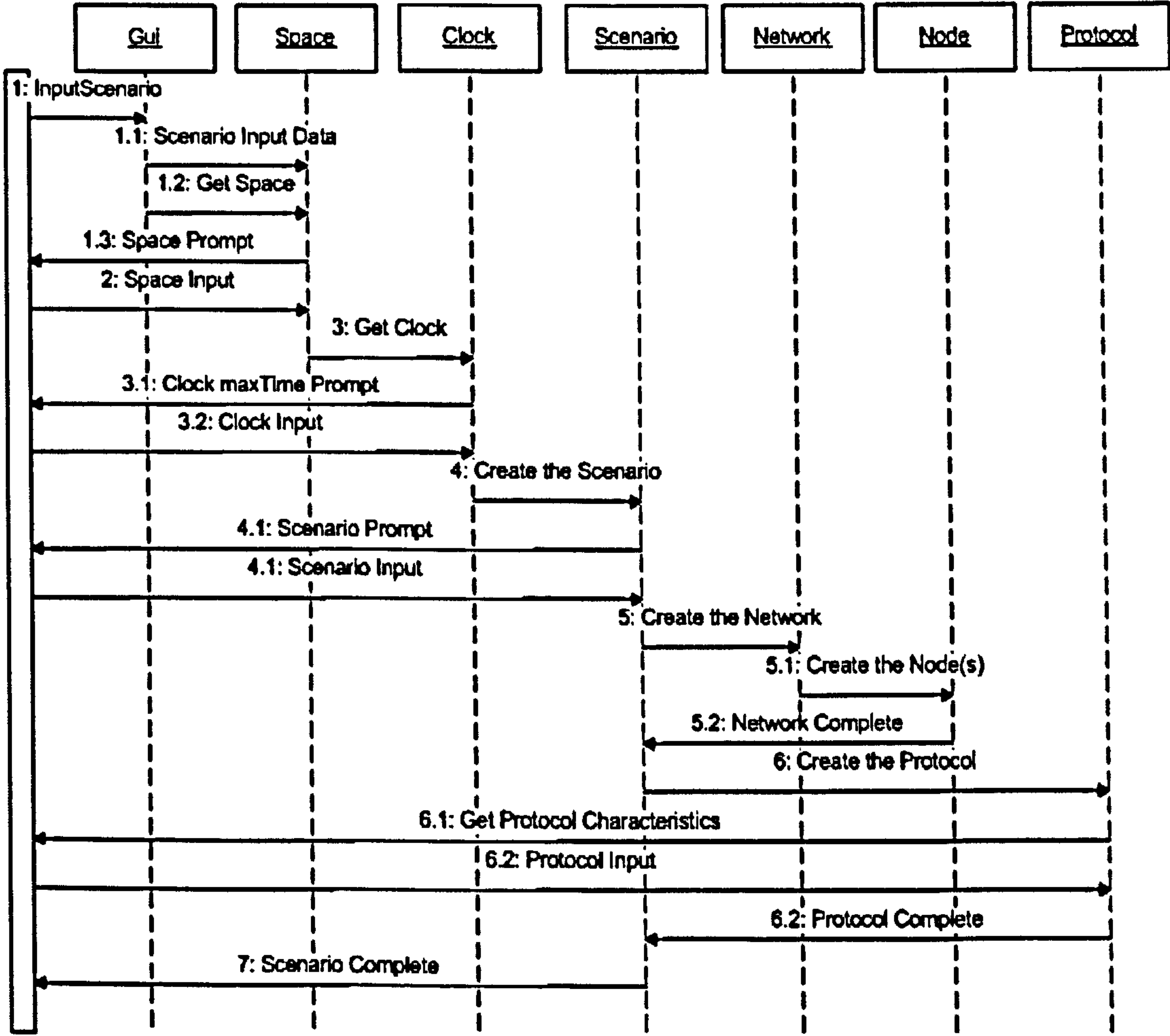


Figure 5.10: The UML sequence diagram detailing the creation of a communication scenario within Swarm.

Once these have been specified, the network with the respective nodes is created. For each node, the mobility model used is also defined. As a final step, the Protocol of communication is specified in terms of its respective attributes.

The second sequence diagram focuses on the assessment of a routing protocol within a specific communication scenario through layer 2 and layer 1 (figure 5.3) of the adaptive component of the system. This is illustrated in figure 5.11.

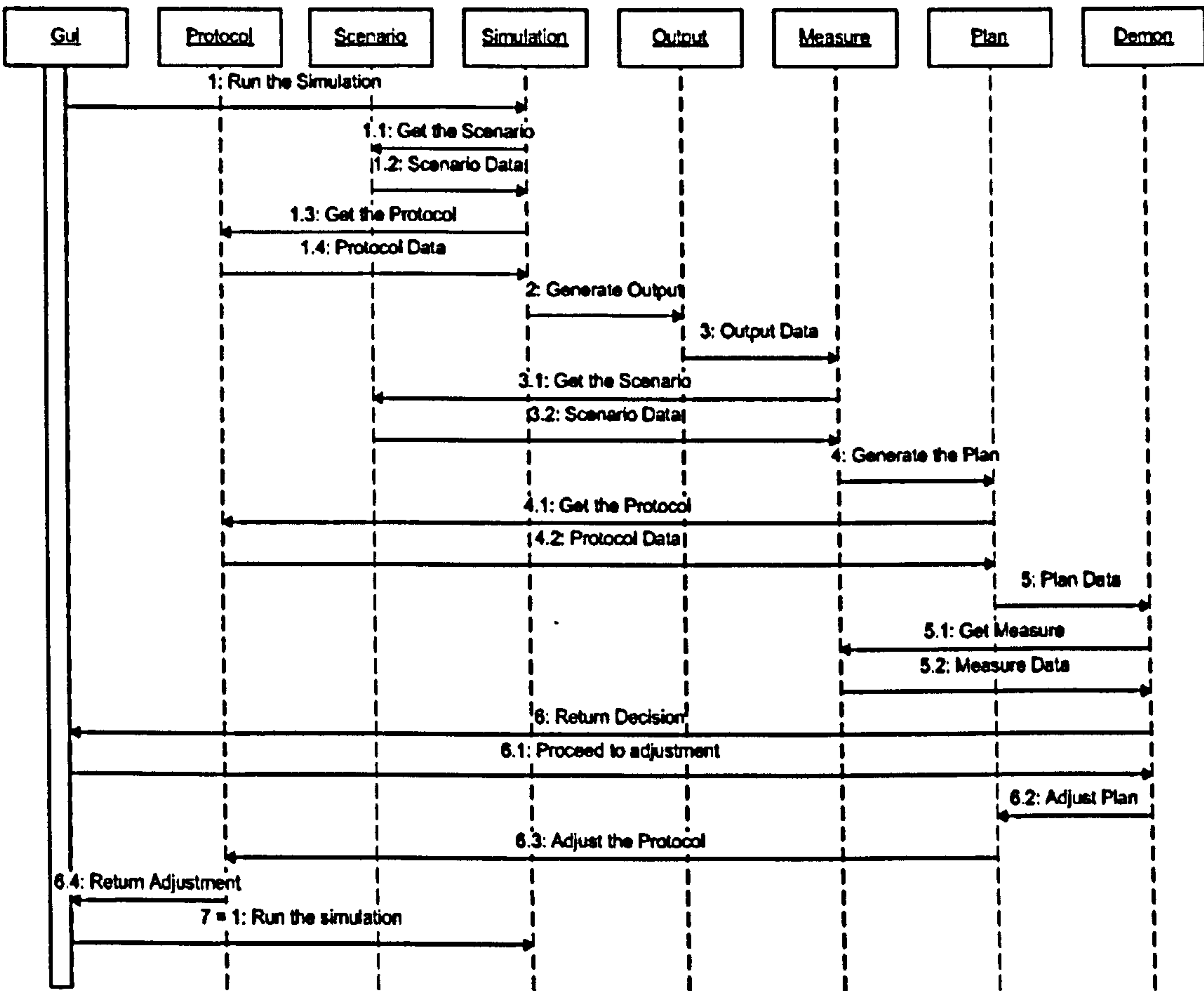


Figure 5.11: The UML sequence diagram detailing the adaptive selection process of a communication protocol.

For the purpose of assessing the Protocol under question, a Simulation of the communication scenario is performed. This simulation produces the corresponding output data, which are held in an Output object. Combing the output data with the communication scenario yields a measure of performance. This Measure triggers the creation of an adaptive plan, which in turn takes into consideration the characteristics that the protocol utilises. The two entities (i.e.

the measure of performance labelled as Measure and the adaptive plan labelled as Plan) trigger the decision demon to make a selection in a similar way to that of a Pandemonium reviewed in section 4.3. Once this process has completed, the assessment of the Demon on the decision level is propagated back to the Plan, which in turn modifies the characteristics of the Protocol accordingly. The attribute values selected are reported back to the Gui and the simulation is executed again, this time under a different protocol specification.

The three classes, namely Demon, Plan and Measure aim to recreate the process of a Pandemonium within the adaptive system. Going back to figure 5.3, each of them inherits two object instances from the layer below it and each has a compute operation for the corresponding characteristic role which they have.

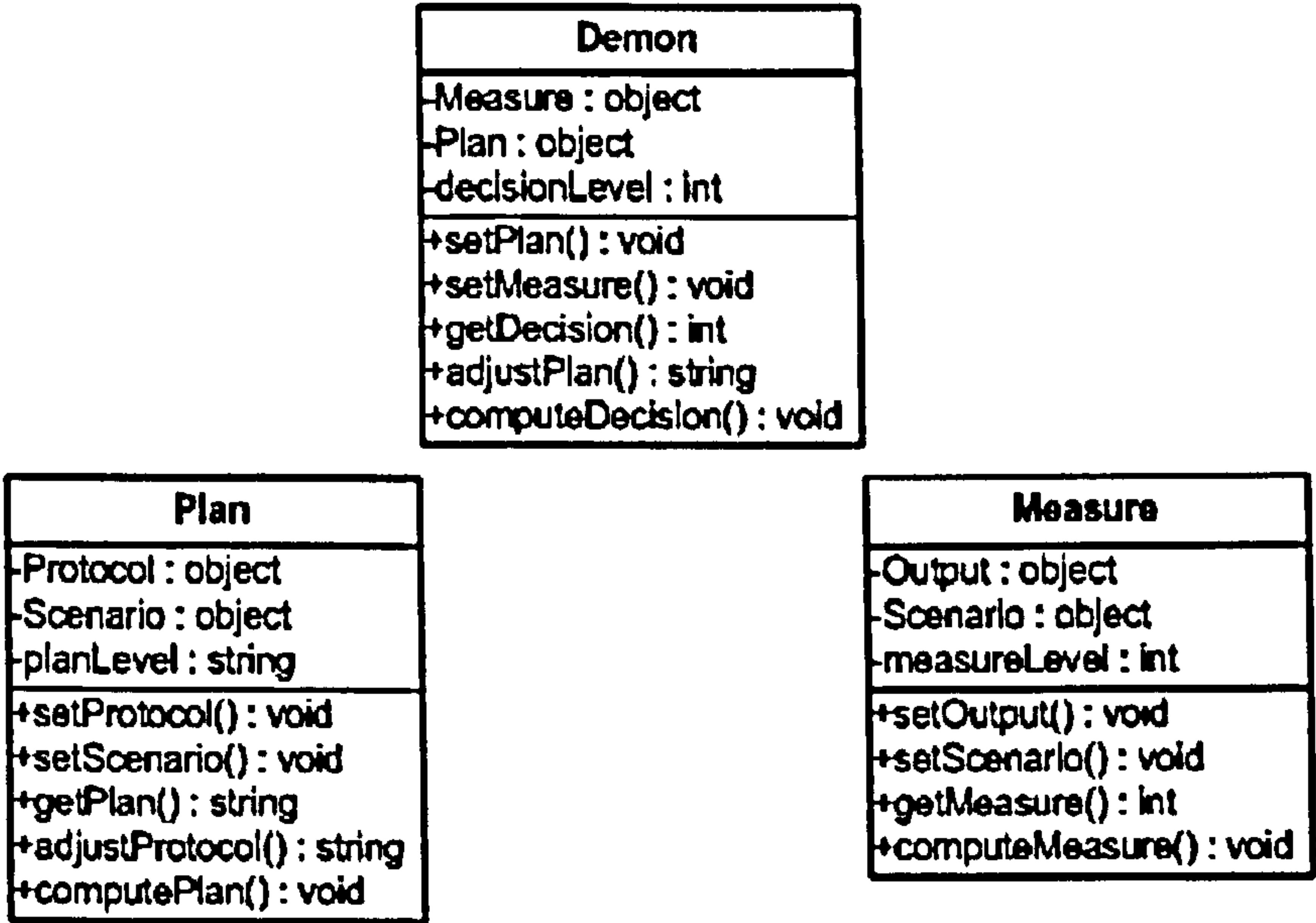


Figure 5.12: The UML class diagrams for the three adaptive classes, namely Demon, Measure and Plan.

Thus, as illustrated in figure 5.12, the decision demon is represented by a `decisionLevel` integer value in the Demon class, in which the final computation is performed (similar to that of level 4 of a pandemonium). The verdict on the fitness of the protocol for the specified scenario is returned through the

getDecision operation. This computation takes into account the difficulty of the malicious attack for the specific scenario and counter-balances this on the adaptive plan. The adjustPlan operator works on the principle that the easier a scenario is, the harder for a protocol to qualify as a candidate. Generally, where success is encountered, small increments are applied to the adaptive plan, while when the success is minor, bigger increments are enforced. The reason why the adjustPlan operator works with strings instead of integers is so that to cover the number of dimensions which characterise a routing protocol, as defined by the attributes of the Protocol class.

Similarly, the Plan class computes the planLevel (again in String format) by taking into account the Protocol as well as the Scenario deployed. A Protocol can be modified accordingly with the adjustProtocol operator, which aims to apply the String representation of the planLevel to the attributes of the Protocol in a similar way to that of the Demon class.

Finally, the Measure class acts as a measure of performance on the Scenario based on the output received through the Output class. For this, the characteristics of an intrusion in an Intrusion Detection System are used, yielding the computation of the measureLevel attribute through the computeMeasure operator. Thus, depending on the difficulty of the attack, the output generated for the simulation illustrates the success or failure of the protocol to handle the threat(s) present within the network.

With the completion of the adaptive descriptions for the respective objects in the dynamic modelling of the system, the next section depicts an overview of the system developed, stating its known advantages and disadvantages from the design that was carried through.

5.7 Swarm: An adaptive protocol selection system

Swarm goes beyond the conventional capabilities of network simulators. Not only does it have the ability to simulate specific MANET communication scenarios in two dimensions, but also it attempts to optimise protocol operation for the given scenario. Coded solely in the Java programming language, a screenshot of the GUI can be seen in figure 5.13 below.

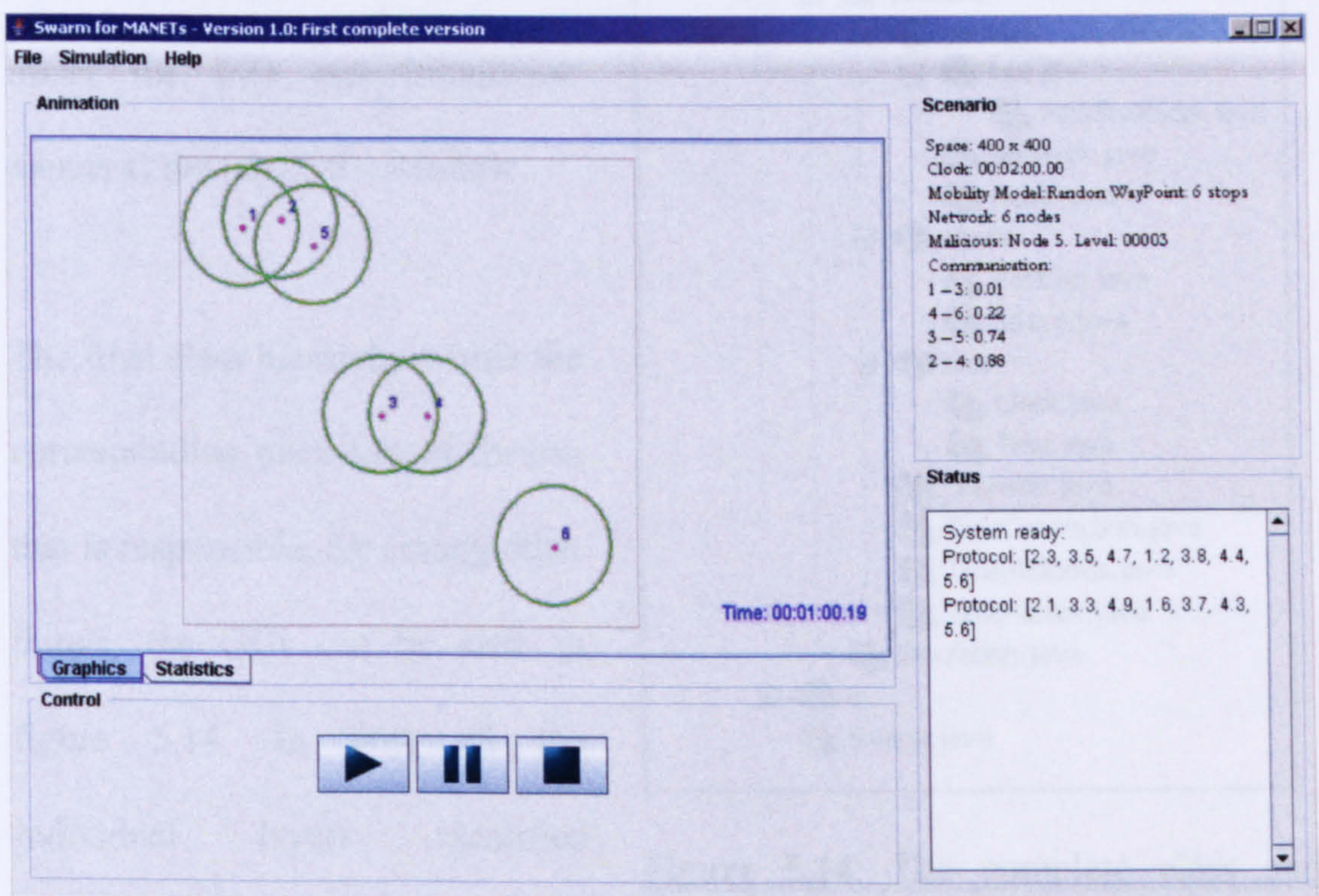


Figure 5.13: The GUI of Swarm consisting of four separate panels.

The user interface of the system is divided into four panels; the first panel, labelled Animation, carries the animation of the simulation scenario, along with the statistics in trace format in a different tab. The second panel, labelled

Scenario, yields all the scenario characteristics of the simulation currently loaded into the system. The Status panel shows the results obtained for each protocol specification, as the adaptive sub-system attempts to find the fittest protocol for the specific case. Finally, the Control panel gives the user the ability to start, pause and stop the simulation. Further options for loading simulation scenarios, as well as adjusting the format in which output is presented in the Status panel can be found under the File and Simulation menus at the top of the window.

The final class hierarchy within the corresponding packages of Swarm that is responsible, for among other things, the GUI can be seen in figure 5.14. In this, all the individual layers identified throughout this chapter, as well as

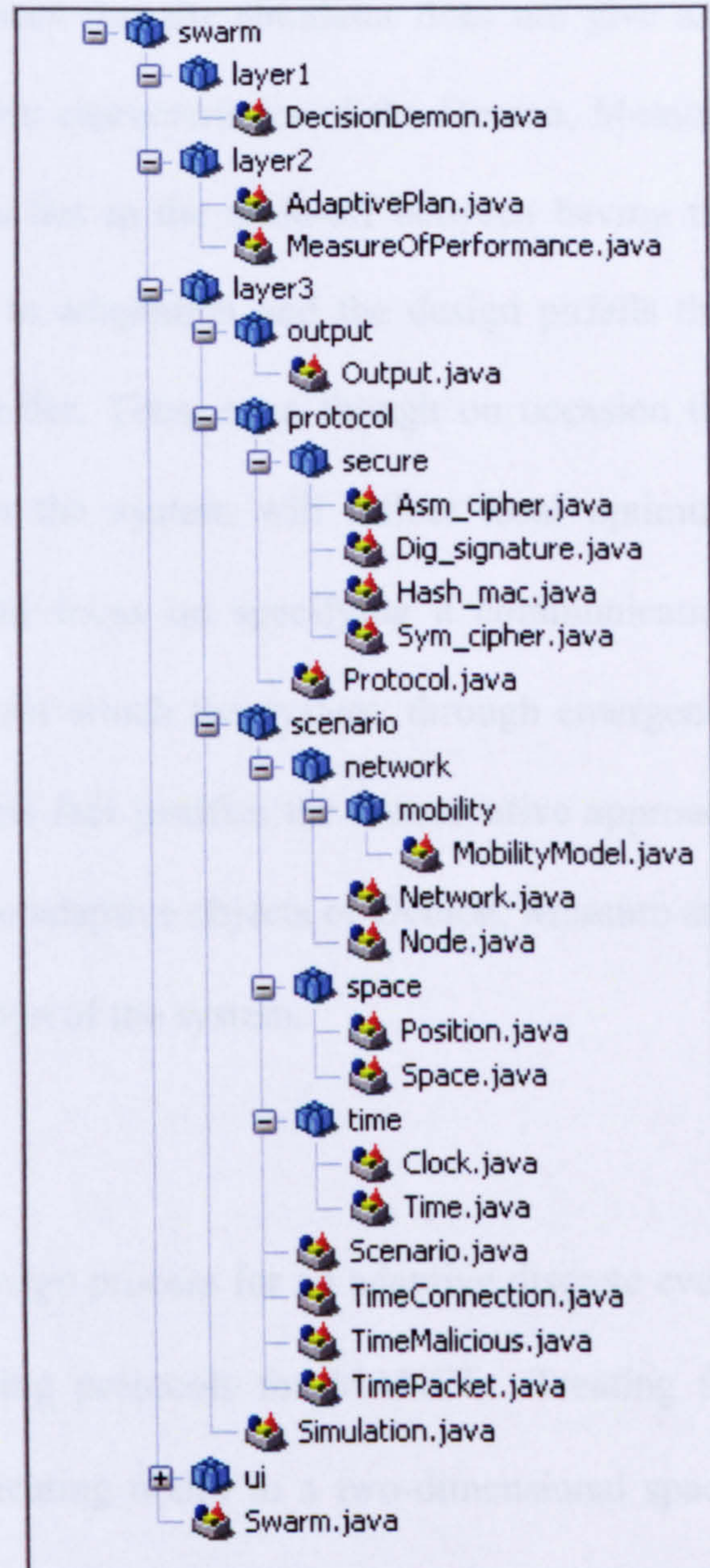


Figure 5.14: The complete class and package hierarchy within Swarm.

entities identified for the communication scenario can be seen. A point worth noting is that the packages network, space and time with their respective sub-packages and classes all fall under the scenario package within layer 3. This illustrates the potential that the Swarm simulator has, to model and attempt to

provide solutions to problem definitions well outside those of malicious intent within ad-hoc networks.

In spite of this, one of the key features that the simulator does not give any control to the user over is the adaptive characteristics of the Demon, Measure and Plan classes. The reason for this lies in the trade-off between having the ability to adjust parameters relating to adaptation and the design pitfalls that such systems (section 4.6) have to offer. Thus, even though on occasion the protocol specification received from the system will reflect local optimum values¹³, any user of the system can focus on specifying a communication scenario through the design process for which the system, through emergence will attempt to find a solution to. This fact justifies the conservative approach towards the compute operations in the adaptive objects of Demon, Measure and Plan, seen in dynamic model description of the system.

5.8 A Summary

This chapter has focused upon the design process for an adaptive discrete event simulator, capable of selecting routing protocols for MANETs. Treating the network as a collection of communicating nodes in a two-dimensional space, our objective has been to allow for the selection of a protocol in an attempt to match the adversarial behaviour present within the network. The sub-system for the protocol generally holds the attributes seen in the classification of routing protocols (described in section 2.1). The sub-system for the scenario deals with

¹³ This can lead to loops in the design process which must be avoided for the system to behave outside the pitfalls and limitations seen in section 4.6.

the communication scenario at hand. Building on the entities identified in adaptive systems modelling (section 4.4) this layer incorporates two sub-systems, namely the adaptive plan and the measure of performance. Consequently, the adaptive plan feeds as a sub-system from information received by the protocol sub-system and the communication scenario undergoing simulation.

Furthermore, we have presented the use case model in the requirement analysis for the system, as well as the way in which the network, scenario and protocol have been modelled. This analysis has embraced both the static and also the dynamic aspects of the modelling process. Finally, the way in which the entities of the system have been ordered, resulting to a number of different classes was put forward.

Building on the system developed, the penultimate chapter puts Swarm to the test, offering a number of results as well as a discussion on the values obtained for specific communication scenarios where one or more nodes have been acting in a malicious manner towards other nodes that are part of the MANET.

Secure protocol designs

6.1	Solutions to the black hole problem
6.1.1	Network and scenario configurations
6.1.2	Results obtained for packet drops
6.1.3	Feasibility analysis and discussion
6.2	Utilising cryptographic primitives within a protocol
6.2.1	Network and scenario configurations
6.2.2	Results obtained
6.2.3	Feasibility analysis and discussion
6.3	Combining protocol characteristics with cryptography
6.3.1	Network and scenario configurations
6.3.2	Results obtained
6.3.3	Feasibility analysis and discussion
6.4	Conclusions

This chapter presents the results obtained in the simulation scenarios tackled by Swarm. Divided into three parts, the first section presents and discusses the results obtained for the security of a protocol as described through its corresponding characteristic pairs. The second focuses on the results obtained in the utilisation of basic cryptographic primitives in the communication process, mainly towards active attacks. The third combines the two, illustrating by example a number of potential protocol designs and their corresponding attributes. To conclude, the last section of this chapter discusses the feasibility of the protocols presented and feasible ways for their implementation.

6.1 Solutions to the black hole problem

One of the fundamental scenarios examined in the communication process of MANETs relates to the ability of nodes to drop incoming packets, carrying a destination address different to that of the node where they are dropped. Despite this being a passive attack, it addresses one of the key attributes of ad-hoc networking; routing information, on demand, via other nodes. As this is a well-studied type of attack, with known solutions on the protocol level [105, 107, 109], we will use it to examine the integrity of the Swarm system. This provides a simple technique for questioning the feasibility of the system, as well as providing a way of calibrating it towards protocol specifications for more complex scenarios.

Section 3.3 defines the attack resulting from a black hole. Such a node always responds to route requests regardless of whether or not it has a valid route to the destination node. Once data packets routed by the source node reach the black hole node, they are dropped. This adversarial model assumes non-cooperating malicious nodes that do not permit partial packet drops that result in “grey” holes. The section that follows describes such an attack in terms of a network and a corresponding communication scenario that can be deployed within Swarm.

6.1.1 Network and scenario configurations

To configure the scenario of communication within the network, we revisit the specification of a typical scenario, as taken into account in sub-section 5.2.1 while designing Swarm. Thus, similarly to other research models [235, 236], we

define a network consisting of 50 nodes, wherein each node utilises the random waypoint mobility model. The maximum available bandwidth on the network is in the range of $[0.5 - 2.0] \text{ Mbs}^{-1}$, selected randomly within this range for each of the 50 nodes. Similarly, the maximum delay that a node can experience on the network is set to be within the range of $[0 - 500] \text{ ms}$, again with each node experiencing a delay selected randomly from this range. Furthermore, the maximum allowed timeout t (measured in ms), which would be acceptable for each protocol request was set to $t_l = 2 \text{ s}$ for nodes which were a single hop away and $t_{max} = 50 \text{ s}$ for the maximum route request timeout.

The 2-D space was considered to be of dimensions $1500 \times 300 \text{ m}^2$ and each node's velocity tangent was selected randomly from the range of $[0 - 20] \text{ ms}^{-1}$. Each node's broadcasting radius was specified to be 200m . The total simulation time was defined to be 120 seconds ; well above twice the maximum route request timeout. Also the sample interval on which events could occur and be monitored on the network was set to be 0.001 seconds .

Within the communication scenario, 10% of the nodes present (i.e. 5 nodes) were considered to be malicious towards the remaining nodes (including each other) throughout the simulation. Their level of maliciousness was defined through their malicious level (table 5.6) and was set for each one to be equal to $\text{MaliciousLevel} = 00003$. Thus, their behaviour on the network was defined on the protocol level as to drop incoming packets, pending any request from other MNs.

As a data communication pattern, like much previous work in evaluating routing protocols [94, 32], we use 20 source-destination pairs, each transmitting at a Constant Bit Rate (CBR) a flow of 4 data packets per second. Each data packet is set to be 512 bytes in size. Thus, having set the source data rate for each node at 4 packets per second, we can say that the application data payload size is 512 bytes per packet.

Finally, available to the protocol specification were only the attributes defined through each characteristic pair. The protocol description could not carry any cryptographic primitives such as symmetric encryption, hashing and so on. In order to allow for the system to emerge with the correct protocol values for the scenario at hand, the simulation was carried out 100 consecutive times, whereby the results on the adaptive entities of the system, were feedback as a starting point for the next simulation.

6.1.2 Results obtained for packet drops

Based on the above setup, the following results were obtained for each of the seven characteristic pairs. Starting from figure 6.1 representing the pair of LSR vs. DVR to figure 6.7 representing the pair of single path vs. multiple paths, the values obtained for each simulation characterise the protocol specification. The system, having selected the initial random value within the specified range, as defined by equation 5.1, progresses to optimise that value based on the simulation scenario at hand. Building on the results of previous simulations, layers 2 and 1, attempt to optimise each characteristic pair over the total number of the simulations undertaken.

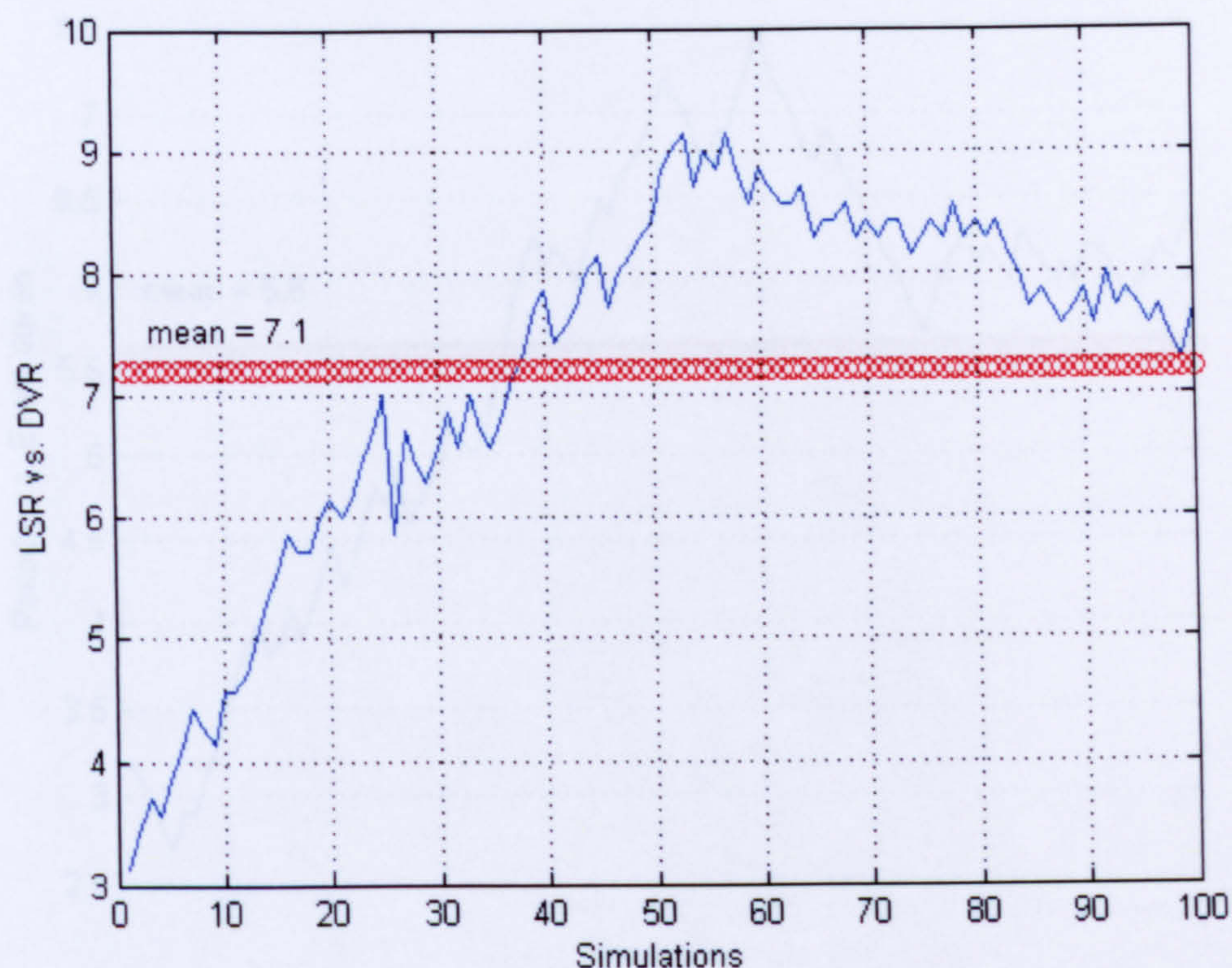


Figure 6.1: Results obtained for the characteristic pair of LSR versus DVR, following 100 simulations of the same black hole scenario.

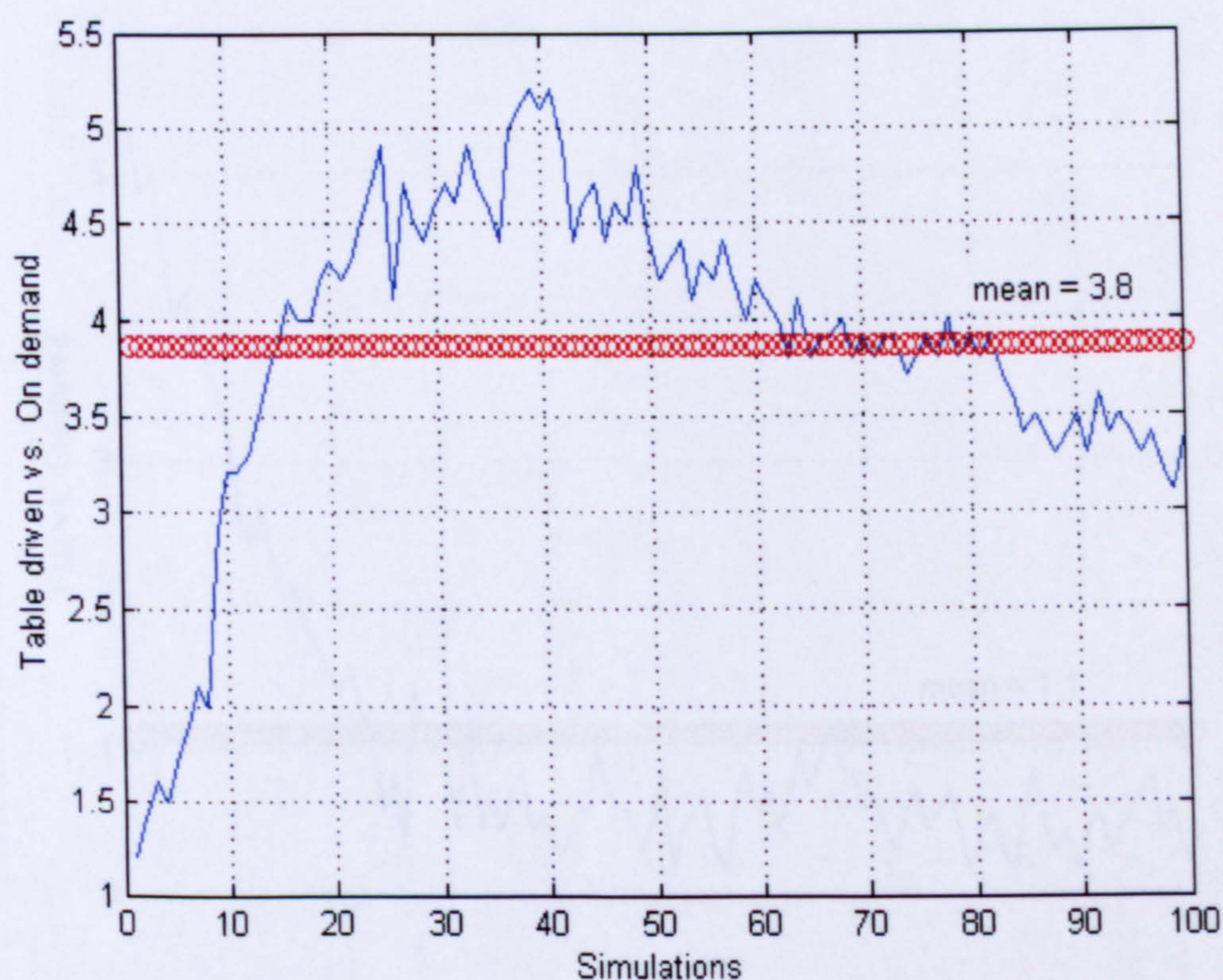


Figure 6.2: Results obtained for the characteristic pair of table-driven versus on-demand routing, following 100 simulations of the same black hole scenario.

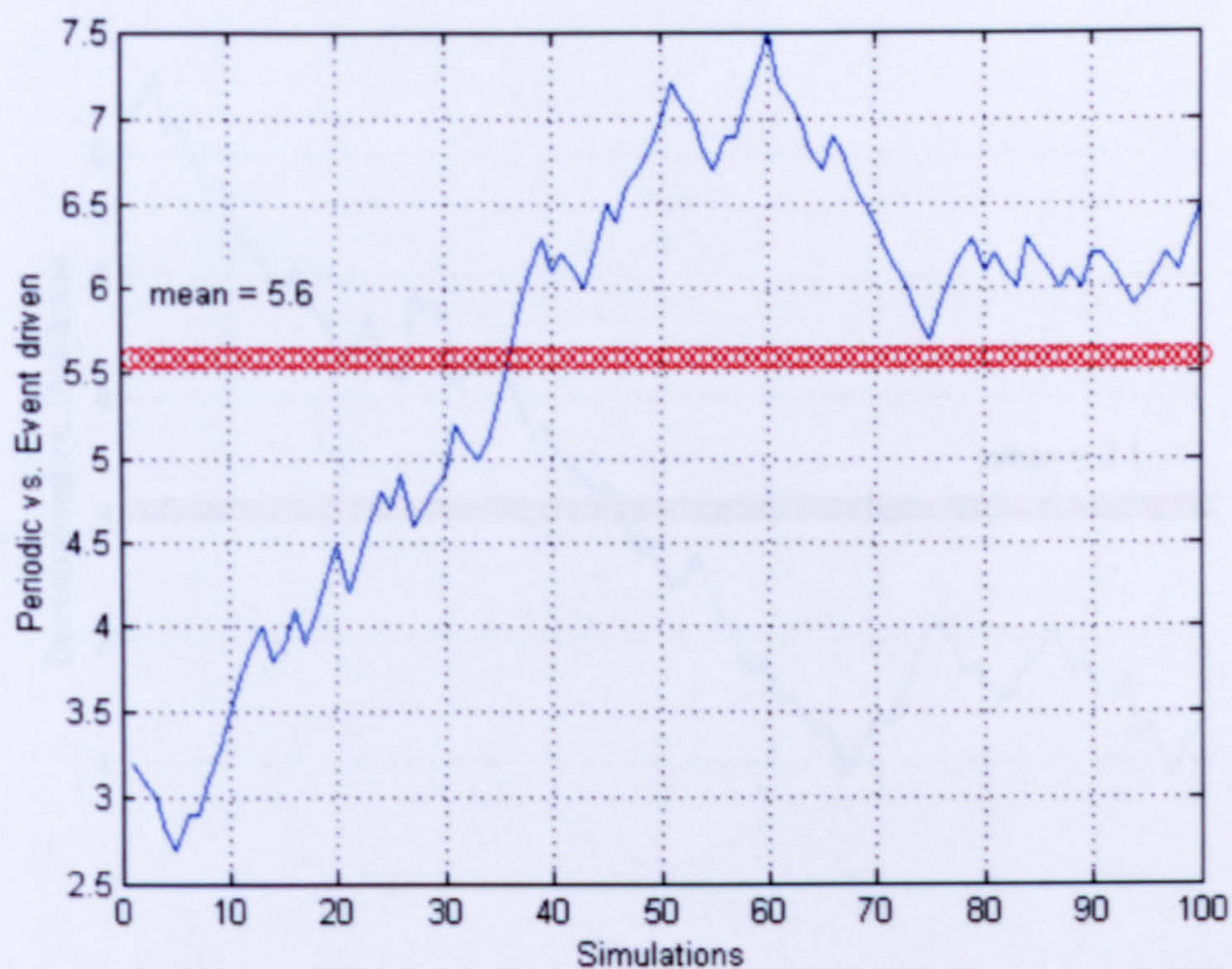


Figure 6.3: Results obtained for the characteristic pair of periodic versus event driven updates, following 100 simulations of the same black hole scenario.

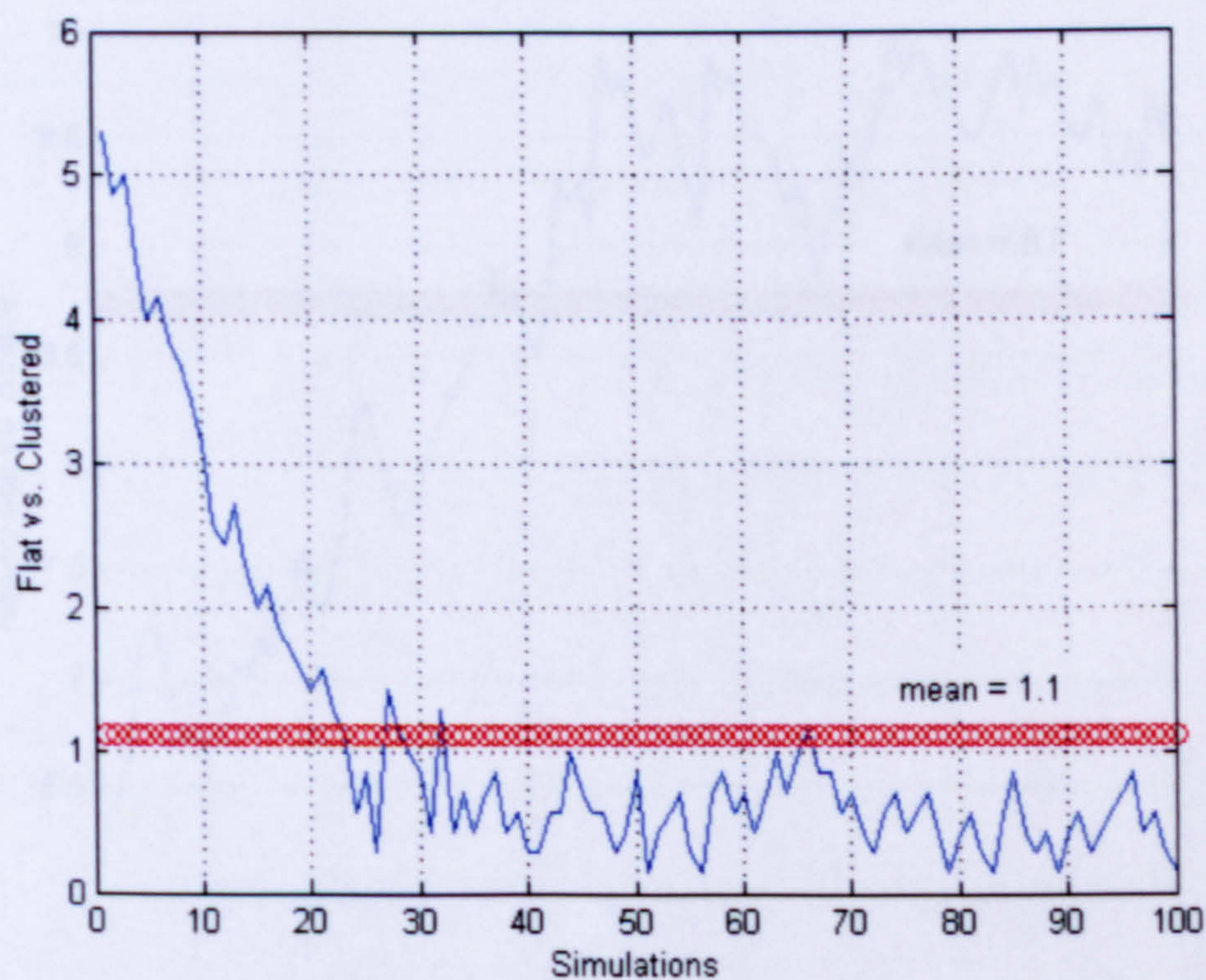


Figure 6.4: Results obtained for the characteristic pair of a flat versus a hierarchical structure, following 100 simulations of the same scenario.

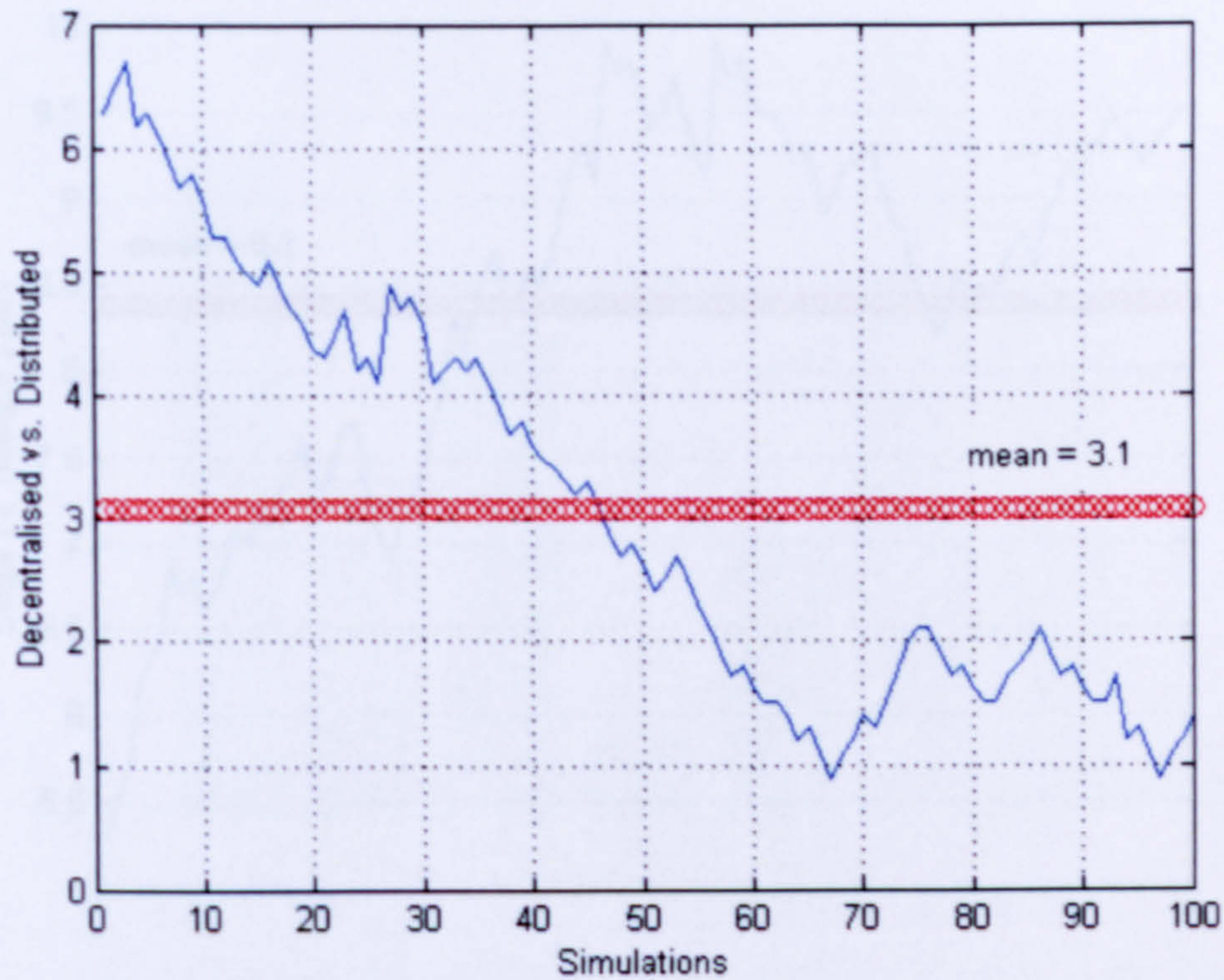


Figure 6.5: Results obtained for the characteristic pair of decentralised versus distributed route computation, following 100 simulations of the same scenario.

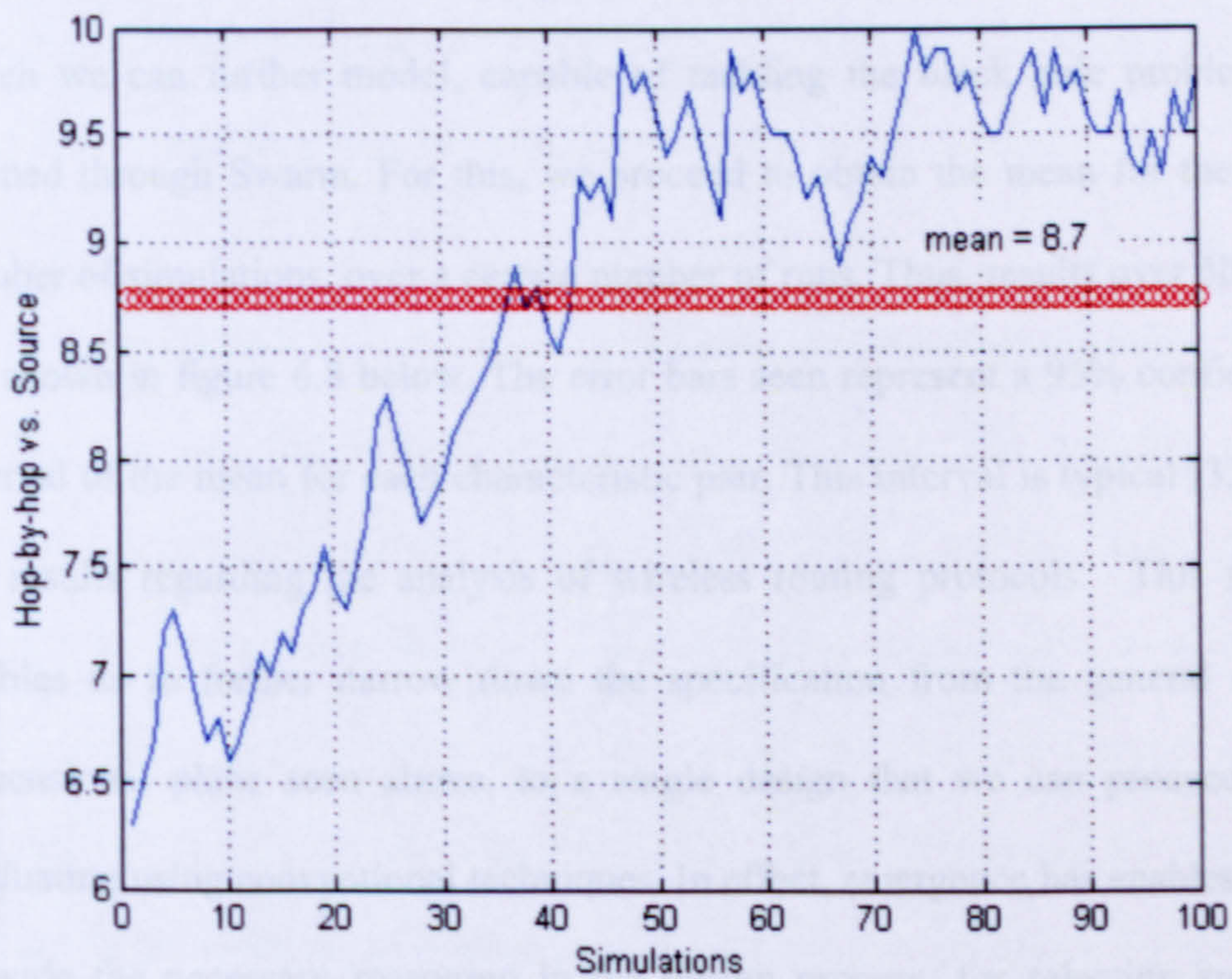


Figure 6.6: Results obtained for the characteristic pair of hop-by-hop versus source routing, following 100 simulations of the same scenario.

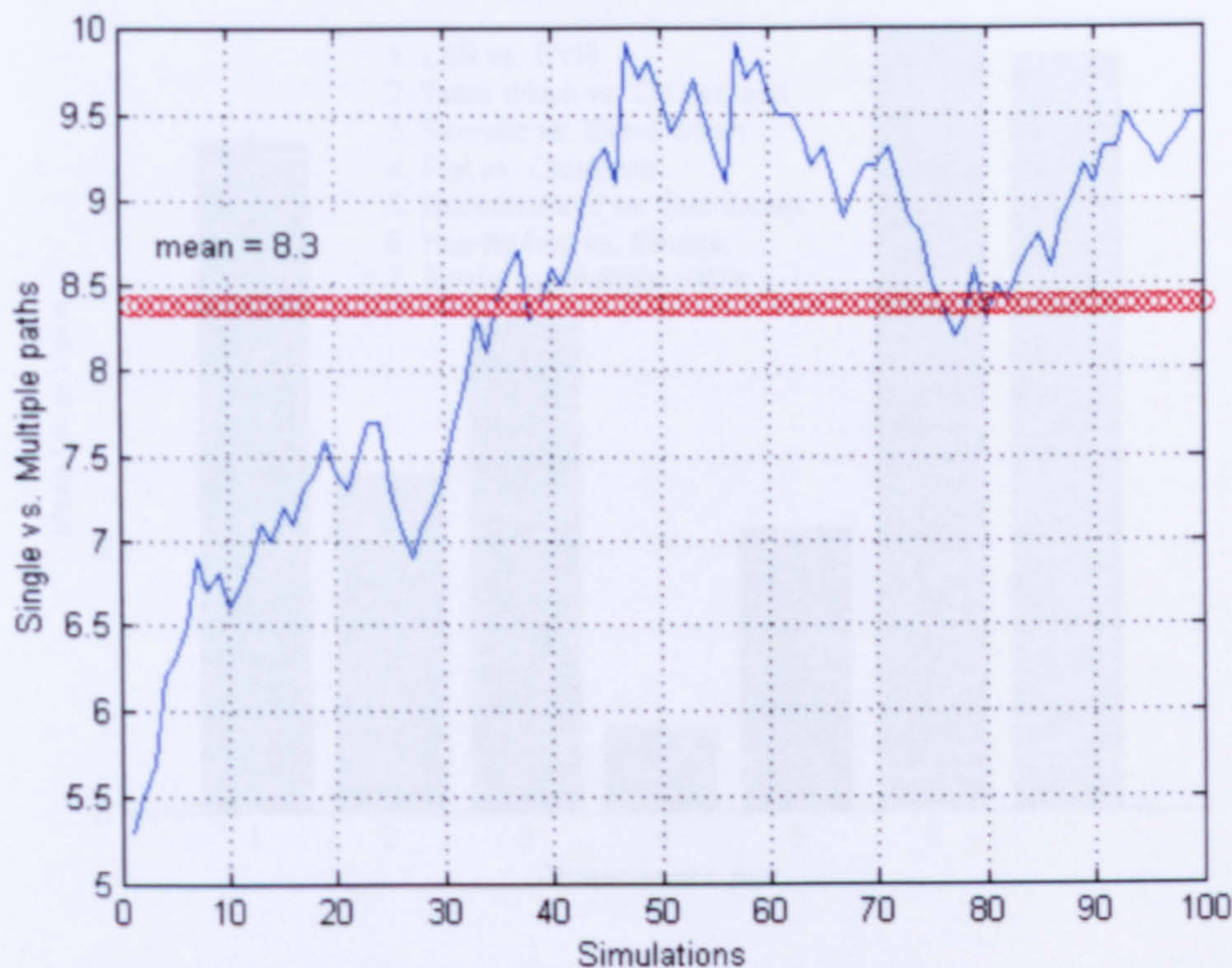


Figure 6.7: Results obtained for the characteristic pair of single versus multiple paths, following 100 simulations of the same scenario.

The results above have as an objective to identify a protocol specification, which we can further model, capable of tackling the black hole problem as defined through Swarm. For this, we proceed to obtain the mean for the total number of simulations, over a certain number of runs. Thus, results over 50 runs are shown in figure 6.8 below. The error bars seen represent a 95% confidence interval of the mean for each characteristic pair. This interval is typical [32, 94] for results regarding the analysis of wireless routing protocols. This figure enables us to further narrow down the specification from the general seven dimensional plane seen above, to a single design that we can proceed into evaluating using conventional techniques. In effect, emergence has enabled us to provide the necessary reasoning in the design process, for selecting specific criteria for each protocol design.

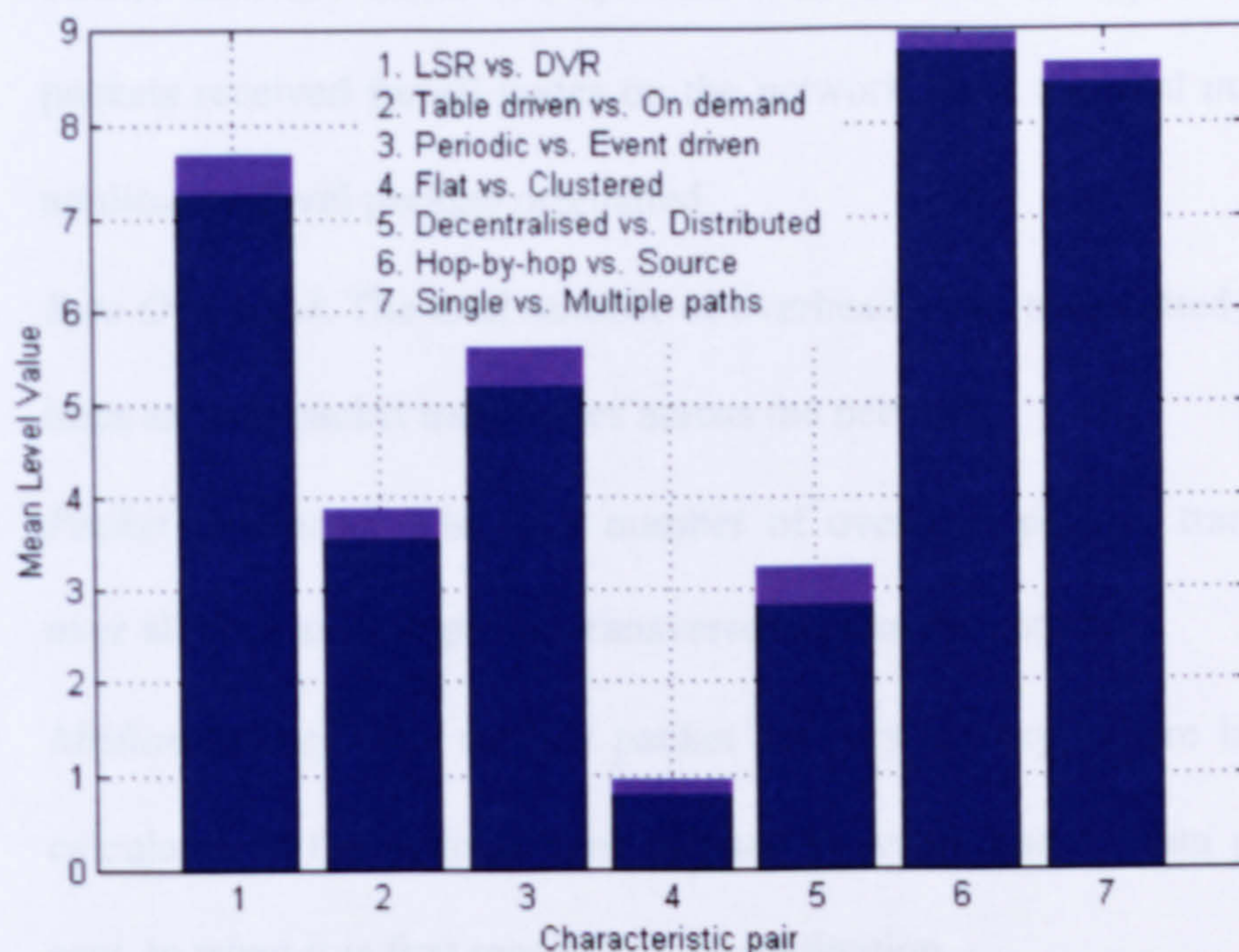


Figure 6.8: The protocol specification, as defined through its characteristic pairs, having being identified as the fittest candidate to tackle the black hole problem.

From this figure and by using equation 5.1, we can reverse engineer a complete protocol specification (using the specification provided in sub-section 5.4.3) and proceed into obtaining a set of more complete benchmark results.

Recalling the standard methods of comparison, presented in section 2.9 and keeping the two dimensional space, broadcast radius and mobility model utilised the same, we proceed as follows. Each node in our simulation moves according to the random waypoint mobility model: a node starts at a random position, waits for a duration called the pause time and then chooses a new random velocity vector of magnitude $[0 - 20] \text{ ms}^{-1}$. Running the protocol evaluation on identical movement and communication scenarios, we measure the performance along four metrics:

- *Packet Delivery Ratio (PDR)*: The total number of application-level packets received for all nodes on the network, over the total number of application-level packets originated.
- *Byte Overhead*: The total number of overhead bytes transmitted, over all hops as each packet transverses across the network.
- *Packet Overhead*: The total number of overhead packets transmitted, over all hops as each packet transverses across the network.
- *Median latency*: The median packet delivery latency, where latency is calculated as the average time elapsed between when a data packet is sent, to when it is first received at its destination.

The performance results presented for each of the above four metrics, take into consideration the standard on-demand routing protocol DSR and are based on simulations over 50 runs this time with the same protocol specification. The error bars presented in each graph within the figure represent the 95% confidence interval of the mean for each of the above metrics.

Figure 6.9 depicts the PDR for the protocol which has the characteristics specified in figure 6.8. We label this as Protocol X and proceed to fully define it through equation 5.1 and each of the descriptions for the protocol characteristics seen in sub-section 5.4.3. As an expectation, we can predict that since Protocol X does not represent a pure on demand implementation, the loss of packets will be greater than that of DSR, as table driven implementations require time for table updates to reach nodes.

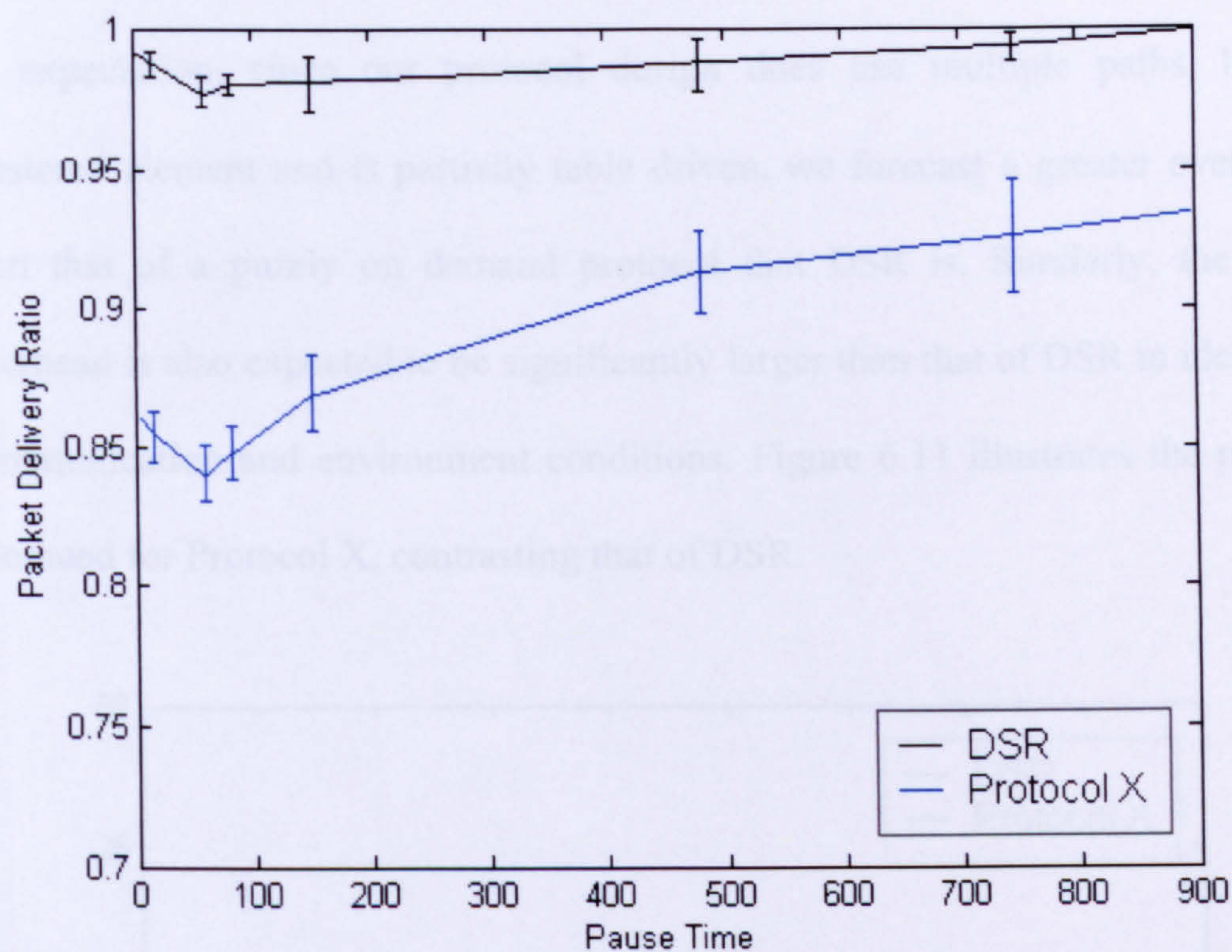


Figure 6.9: The packet delivery ratio for Protocol X and also DSR versus the pause time.

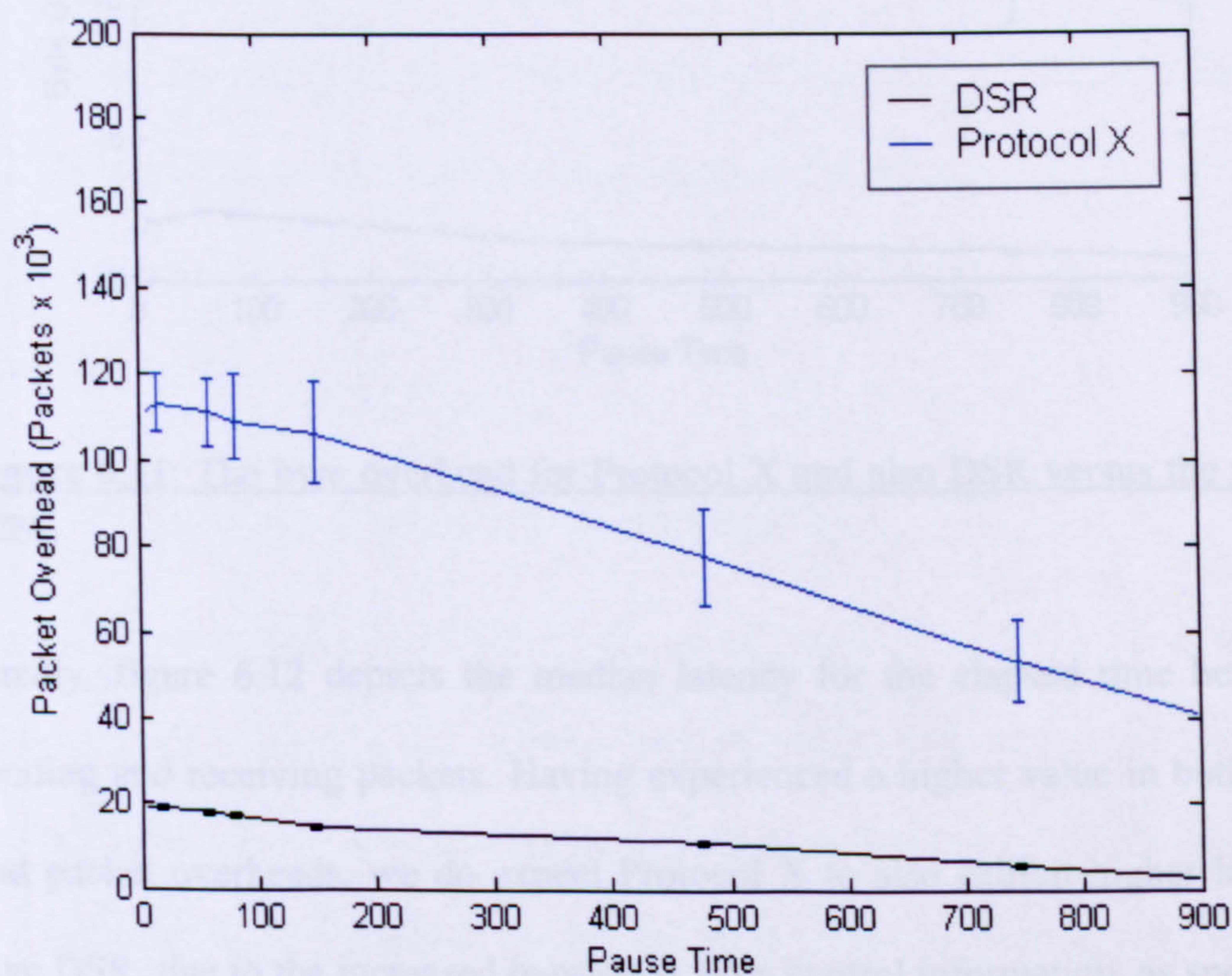


Figure 6.10: The packet overhead for Protocol X and also DSR versus the pause time.

Figure 6.10 above details the packet overhead for Protocol X and also DSR. As an expectation, since our protocol design does use multiple paths, has a clustered element and is partially table driven, we forecast a greater overhead than that of a purely on demand protocol that DSR is. Similarly, the byte overhead is also expected to be significantly larger than that of DSR in identical communication and environment conditions. Figure 6.11 illustrates the packet overhead for Protocol X, contrasting that of DSR.

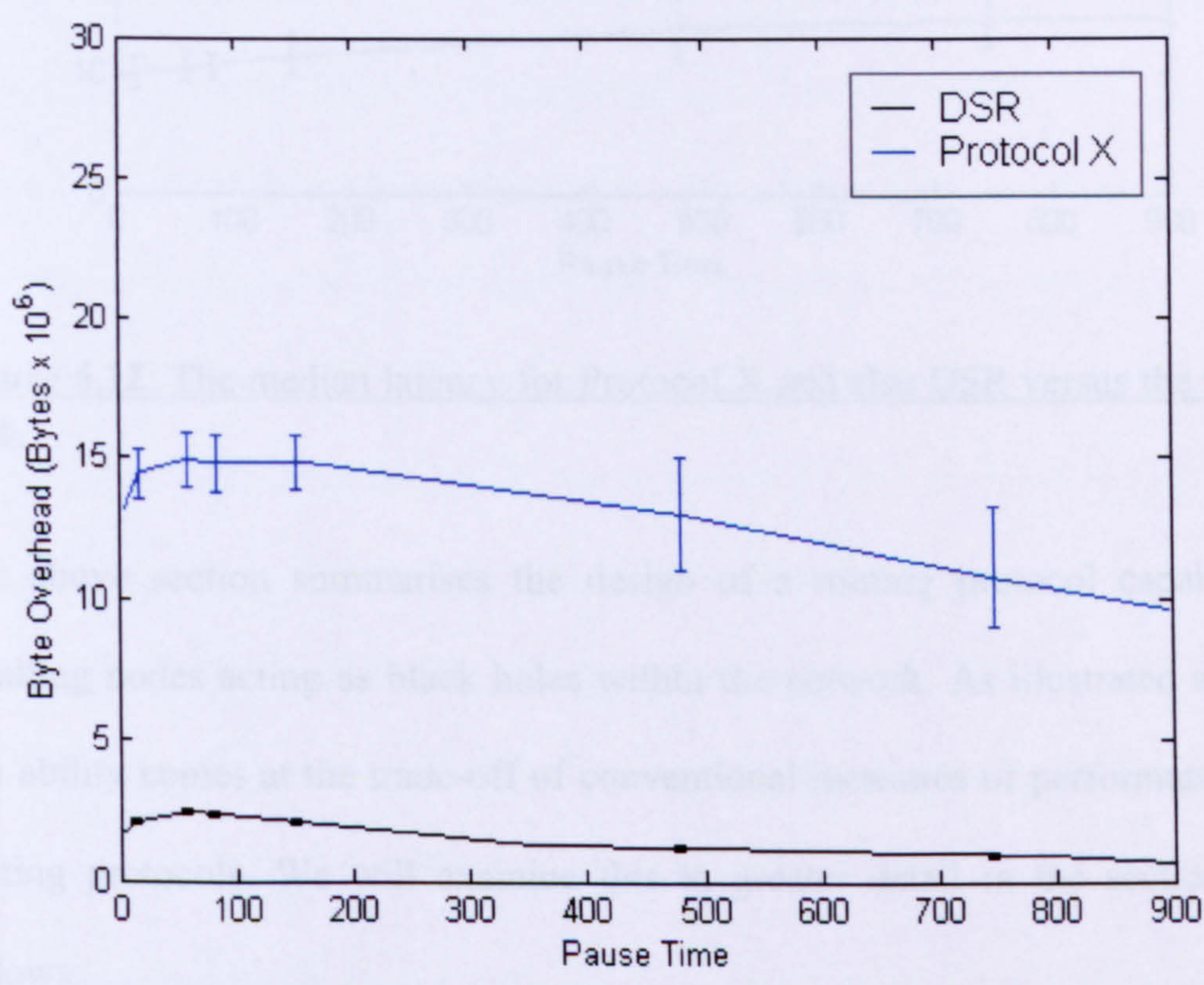


Figure 6.11: The byte overhead for Protocol X and also DSR versus the pause time.

Finally, figure 6.12 depicts the median latency for the elapsed time between sending and receiving packets. Having experienced a higher value in both byte and packet overheads, we do expect Protocol X to also exhibit higher latency than DSR, due to the increased overheads from control information, as specified for each protocol characteristic.

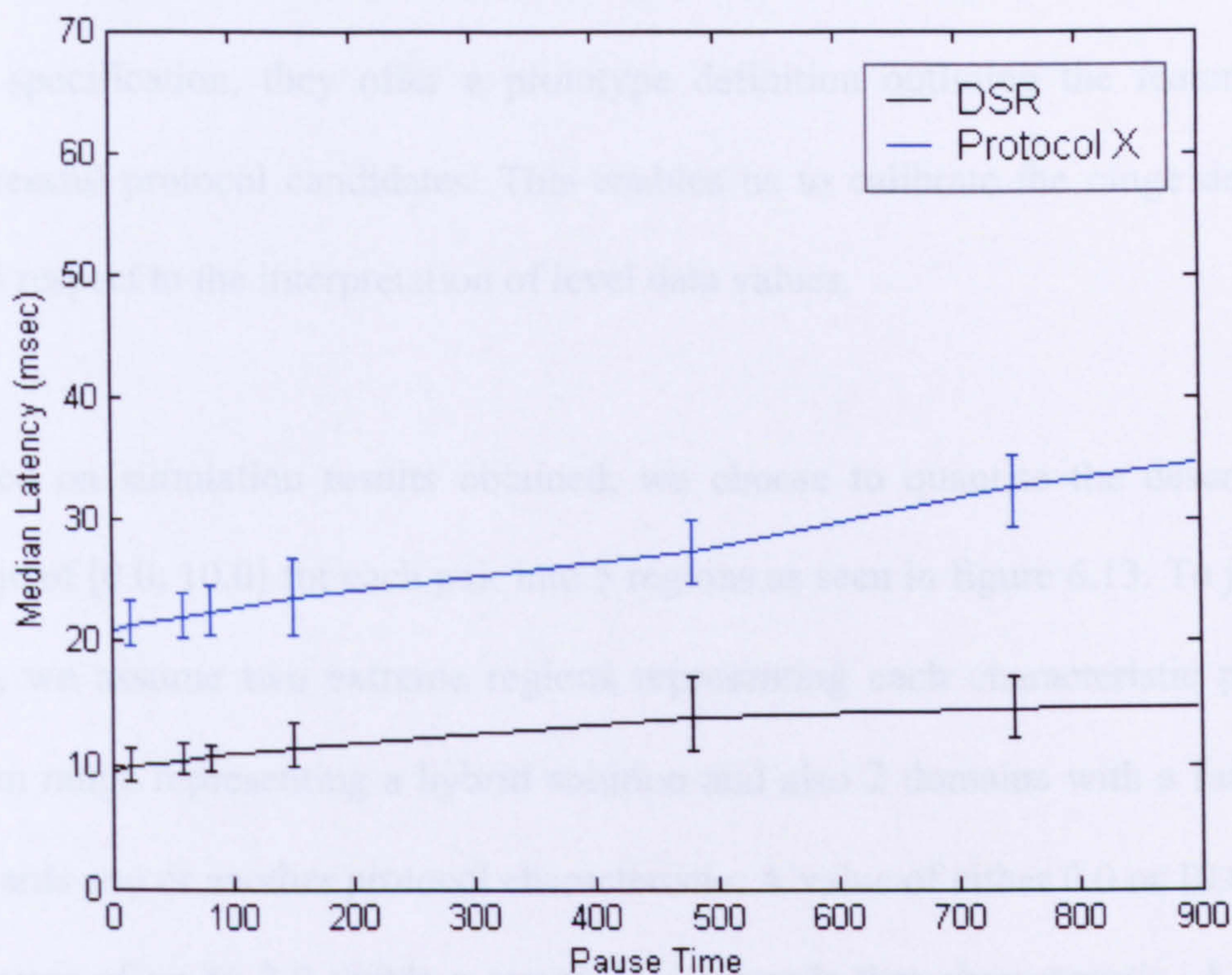


Figure 6.12: The median latency for Protocol X and also DSR versus the pause time.

The above section summarises the design of a routing protocol capable of handling nodes acting as black holes within the network. As illustrated above, this ability comes at the trade-off of conventional measures of performance for routing protocols. We will examine this in greater detail in the section that follows.

6.1.3 Feasibility analysis and discussion

Security against passive eavesdropping on the protocol level comes at a cost. This cost can be seen in the decrease of performance with respect to the PDR, overhead and latency at any pause time. Despite of this being the case, Protocol X does manage to safeguard against the black holes present on the network, continuing to deliver packets via other routes.

Even though the results presented in figures 6.1 to 6.7 do not correspond to a full specification, they offer a prototype definition outlining the features of successful protocol candidates. This enables us to calibrate the range defined with respect to the interpretation of level data values.

Based on simulation results obtained, we choose to quantise the descriptive range of $[0.0, 10.0]$ for each pair into 5 regions as seen in figure 6.13. To justify this, we assume two extreme regions representing each characteristic pair, a mean range representing a hybrid solution and also 2 domains with a fair bias towards one or another protocol characteristic. A value of either 0.0 or 10.0 with an error of up to 2.0 yields a strong bias towards that characteristic. A value close to the mean with an error of up to 1.0 yields a necessary hybrid approach for that characteristic pair. Finally, for the range of $[2.0, 4.0]$ and $[6.0, 8.0]$ we consider a bias towards the characteristic closer to that range, without eliminating the rare occurrence of the opposite behaviour. For this we consider that the closer the value is to 2.0 and 8.0 respectively, the smaller the probability for that occurrence to take place.

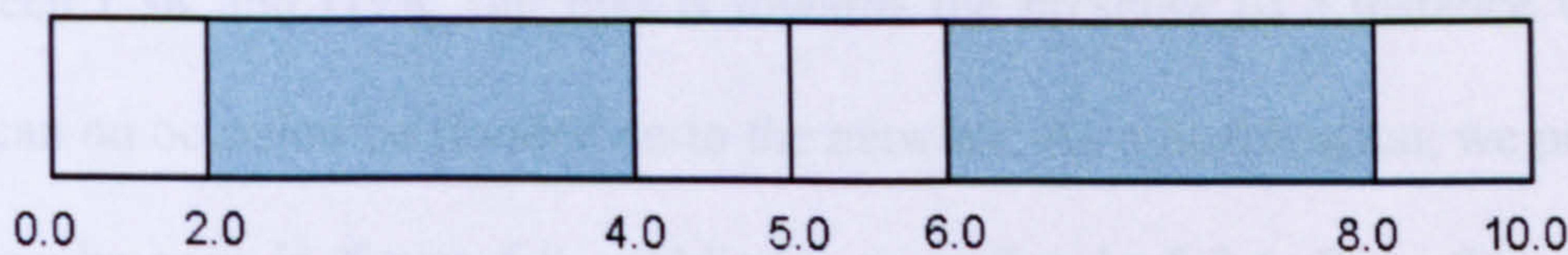


Figure 6.13: The selection range used for each characteristic pair.

Thence, we first review characteristics within the limit values of $[0.0, 2.0]$ and $[8.0, 10.0]$. From the results obtained for the characteristic pair of a flat versus a clustered structure seen in figure 6.4, we select a protocol that has a flat structure where every node has the same rights across the network. As seen in

the figure, despite selecting a random value close to the mean of the range as an initial condition, our adaptive system closely decreases that value to the range between $[0.0, 2.0]$ providing a mean of 1.1 after the total number of simulations has been performed.

With similar results for another two characteristics and based on the definition of our measure of performance, we also prefer a protocol that takes into account multiple paths acting on as much feedback as possible. Figure 6.7 illustrates the average level obtained from 20 simulation sets for all the characteristic pairs for which individual simulation results have not been presented. Using multiple paths is particularly useful for the re-routing of packets knowing that a black hole is present within the MANET. Finally, with respect to source routing, we treat a packet as a stand alone entity carrying all the necessary information in order to reach its destination. This justifies the average value of 8.7 (figure 6.6) computed for 20 simulation sets regarding this particular characteristic pair.

With respect to characteristics that appear to be more hybrid in choosing between LSR and DVR, our bias is towards the presence of a distance vector that can on occasion be flooded on to the network. As a justification, we present the results seen in figure 6.8, yielding a mean level of 7.4. Even though the initial condition in this case is closer to the definite range of link state routing, during the course of the simulation of the same scenario the value shifts towards DVR recommending an update of routing tables within the network. As a result, we conclude that the protocol candidate should not flood the network upon a

noticeable change in topology, but should instead encourage random updates of the routing table between neighbouring nodes.

The remaining three characteristic pairs (figure 6.8) also follow a hybrid form. For the case of table-driven versus on-demand routing, our results encourage each MN to hold a local view of the network (by not neglecting any neighbouring information that passes through them) and enquire further about routes to more distant nodes using an on demand approach. Furthermore, for the periodical versus event driven updates of routing information, our results promote a more weighted exchange, where both periodic as well as event driven updates are utilised. Finally, for the pair of decentralised versus distributed route computation, our results show a preference towards the distributed approach, stating that the entire route should not be computed within a single node.

Using section 5.4.3, we further simulate this time the identical protocol scenario, with respect to the four metrics selected. Namely, the delivery ratio, the overhead and the average latency; furthermore we take into account the performance of another routing protocol (in this case DSR) for comparison. This approach for evaluating Protocol X, can also be seen for the Ariadne protocol specifications in [94].

From this, we reach the conclusion, that despite having a protocol capable of tackling the black hole problem, as defined in the communication scenario above, its performance with respect to conventional metrics appears to be far worse than that of a well known on-demand protocol. This is completely

justifiable; in order to tackle a security threat on the network, the protocol is required to have available and also circulate more information regarding the routing process. Through emergence, we have the opportunity to optimise the amount of information that is further circulated by not over-specifying particular attributes within the given characteristic pairs. As a result, we can see from figures 6.9 to 6.12 that despite Protocol X being outperformed by DSR in all four metrics, the results for Protocol X are of the same order of magnitude for the metrics observed. This leads us to conclude that the above protocol can be implemented to tackle the black hole problem within MANETs.

The above process for selecting a protocol candidate stems from attempting to solve the black hole problem in the presence of a passive malicious node within the network. As we can see from the results, our adaptive system does not attempt to provide a single solution based on a local property [6], such as limiting the reply packets that can be sent between nodes. Instead, it looks for the global characteristics within a protocol specification and attempts to eliminate ones that do not assist in the problem solving technique.

6.2 Utilising cryptographic primitives within a protocol

Having seen the ability of Swarm to provide protocol designs that can safeguard against passive malicious intent on the network, we proceed into expanding this functionality to incorporate cryptographic primitives. As detailed in section 3.3, there exist a number of active attacks where the malicious node attempts to manipulate or actively inject network traffic with the objective of compromising the flow of information between nodes. We examine a variety of these attacks in the section that follows, defining them as scenarios in Swarm.

6.2.1 Network and scenario configurations

To configure the scenario of communication on the network, we specify a setup containing the same 2D space, nodes and physical properties as that in section 6.1.1. Again, 10% of the nodes were considered to be malicious towards the remaining nodes (including each other) throughout the simulation. Their level of maliciousness (table 5.6) is elevated from that of passive eavesdropping but does not take into consideration any collusion of power between them. Thus each malicious MN is set to have a *MaliciousLevel* within the range of $[00004 - 00009]$, selected at random.

Similarly to the previous section, we use 20 source-destination pairs, each transmitting at a Constant Bit Rate (CBR) a flow of 4 data packets per second. Each data packet is set to be 512 bytes in size. Thus, having set the source data rate for each node at 4 packets per second, we can say that the application data payload size is 512 bytes per packet. These are also the setting for the 5 adversarial nodes present.

Building on an existing protocol, we choose DSDV (section 2.3), which despite being considered to be simple resolving any issue of routing loops, has well documented vulnerabilities [154, 250]. Using the secure protocol package (figure 5.14 & sub-section 5.4.3) we will attempt to safeguard the performance of DSDV in the presence of active, non-colluding attackers.

In terms of security, we make the following assumptions. Each node has a private and public key pair, with every node, having knowledge of every other

node's public key. Any hash algorithms used (figure 3.8) are non-keyed and operate on the message or in this case packet alone. The signing of any message on the packet level, takes place using the node's private key, which is kept secret. Furthermore, the existence of a symmetric key between every node pair is also postulated. As seen in [166] this is not an unrealistic requirement.

Finally, in order to target operations of active adversarial behaviour, from the four methods of communication that a node has on the packet level, namely, sending, receiving, dropping and forwarding packets, we allow Swarm to redefine two. Thence, for both the send and receive methods that involve routing table dumps being transmitted and applied between nodes as control information, each node has the ability to apply any of the four security classes available. Thus, $(4 \times 3 \times 2) \times 2$ pairs surface, each giving different abilities to the receiving packet information, as described by the four classes within the secure package.

The success of each of the candidates was measured with respect to the level of intrusion detection achieved within the protocol. This, as defined in sub-section 5.4.3, allowed for 5 different levels of intrusion detection, with *not realising the occurrence of a malicious event* being the lowest and *isolating any information originating from a malicious node* being the highest. To allow Swarm to improve on stated results and avoid local maxima, it was always assumed that a malicious event was taking place within the network. As the protocol used does not incorporate any intrusion detection techniques, the ability to ignore routing information, provided it was though to be invalid was implemented within

DSDV. This gave the system, the ability to notice and achieve 4 out of 5 intrusion detection states, while always aiming for the fifth, namely, *isolating information originating from a malicious node*, despite not being able to achieve it.

6.2.2 Results obtained

The results obtained for DSDV focused on partial and full routing table dumps, which represent the way in which this protocol exchanges control information between nodes. We selected to perform the simulation for 100 runs; each of which returned a secure alteration to the protocol. From this, the decision daemon in layer 1, returned the following preferences, as depicted in figure 6.14. This was derived for the sending process and reversed for the receiving process.

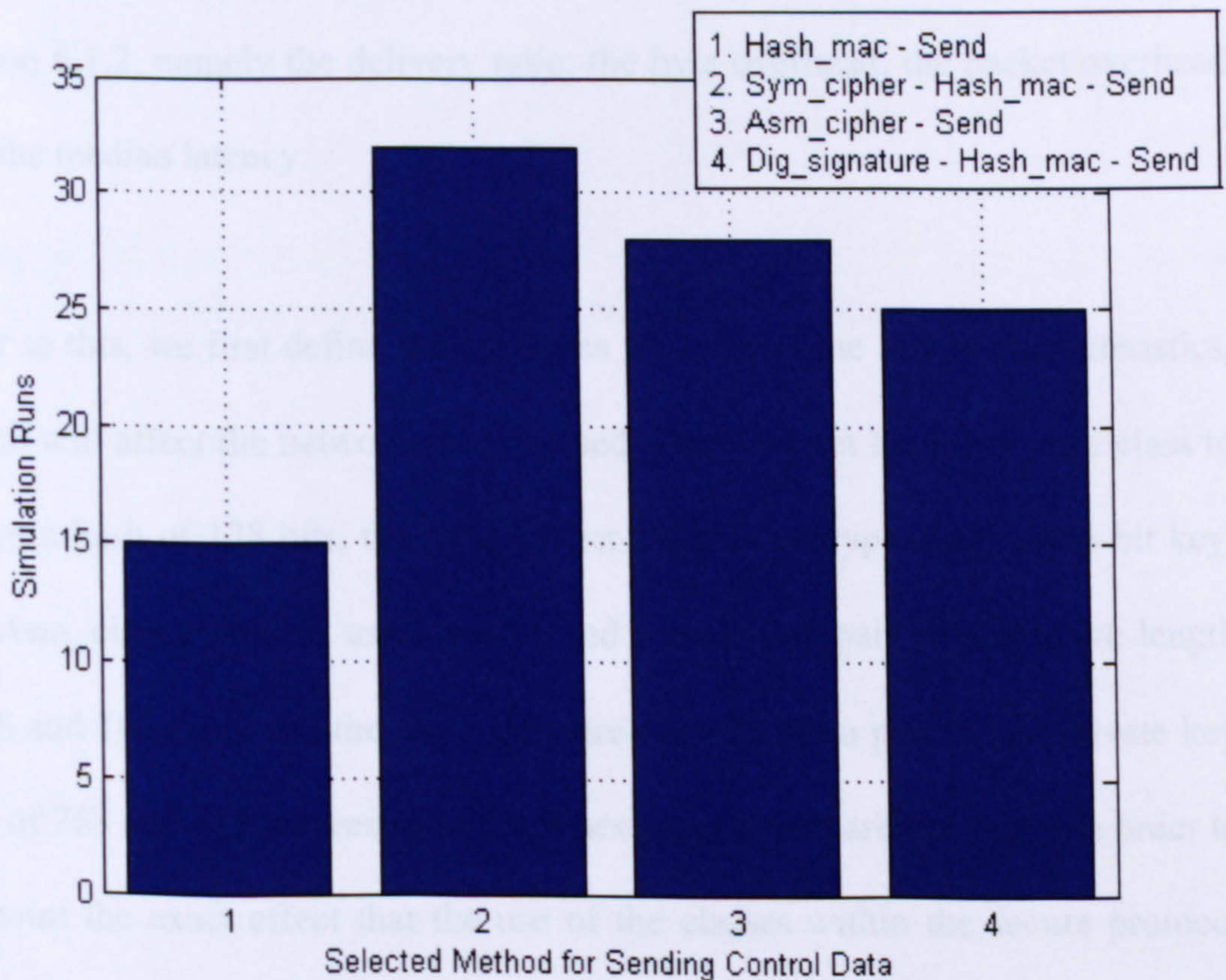


Figure 6.14: The four different protocol methods, as selected by Swarm, for the transmission of control packets within DSDV.

From this, we can derive that the preferred technique for transmitting control data using DSDV is that of symmetrically encrypting each routing table update, obtaining and attaching a hash to it and then sending it across the network. Second comes the process of encrypting it asymmetrically and then sending it, while third comes the process of digitally signing the data, obtaining and attaching a hash and then transmitting them across the network. The final process depicted in the figure is that of obtaining the hash value, attaching it to the data and transmitting it across the network. Following the above results, we proceed into defining each of the four classes on which the above results are based, in order to be able to further simulate the characteristics seen here within.

For each of the above four processes, we proceed into obtaining a comparison with respect to DSDV and each other, based on the four metrics defined in section 6.1.2, namely the delivery ratio, the byte overhead, the packet overhead and the median latency.

Prior to this, we first define the attributes for each of the secure characteristics, which will affect the network metrics used. Thus, we set the Hash_mac class to return a hash of 128 bits, the Sym_cipher class to encrypt using a 256 bit key, the Asm_cipher class to use a public and private key pair of respective length 2048 and 1024 bits and the Dig_signature class to use a public and private key pair of 768 and 512 bits respectively. These parameters are necessary in order to pinpoint the exact effect that the use of the classes within the secure protocol package will have on the derived protocol specification.

The performance results presented are based on simulations over 50 runs this time with the same DSDV protocol specification. The error bars presented in each graph within the figure (as in section 6.1.2) represent the 95% confidence interval of the mean for each of the above metrics.

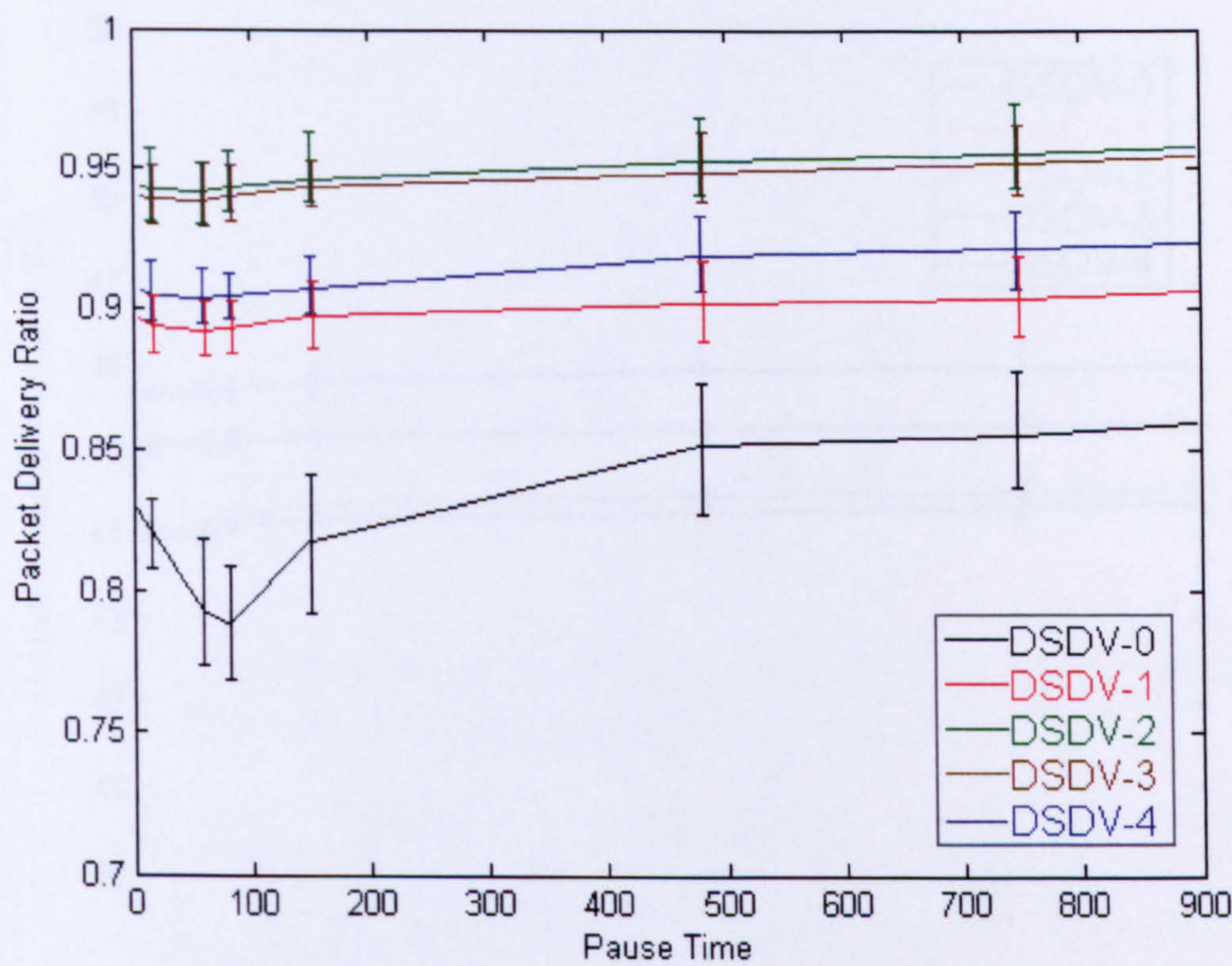


Figure 6.15: The packet delivery ratio for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.

Figure 6.15 above depicts the packet delivery ratio of the original protocol specification of DSDV, labelled as DSDV-0 and also the four secure implementations of it, using the index of figure 6.14. Thus, DSDV-1 represents the protocol that transmits the hash of the routing table update and DSDV-2 the protocol that transmits the routing table, encrypted using symmetric encryption and also hashes the result. DSDV-3 is the protocol that encrypts through a public key the routing table update, while DSDV-4 the protocol that signs and hashes the routing table update transmitted.

Figures 6.16 and 6.17 illustrate the packet and byte overhead respectively, for each of the above secure designs of DSDV. As an expectation, both metrics should be much higher, as more data are being exchanged due to cryptography.

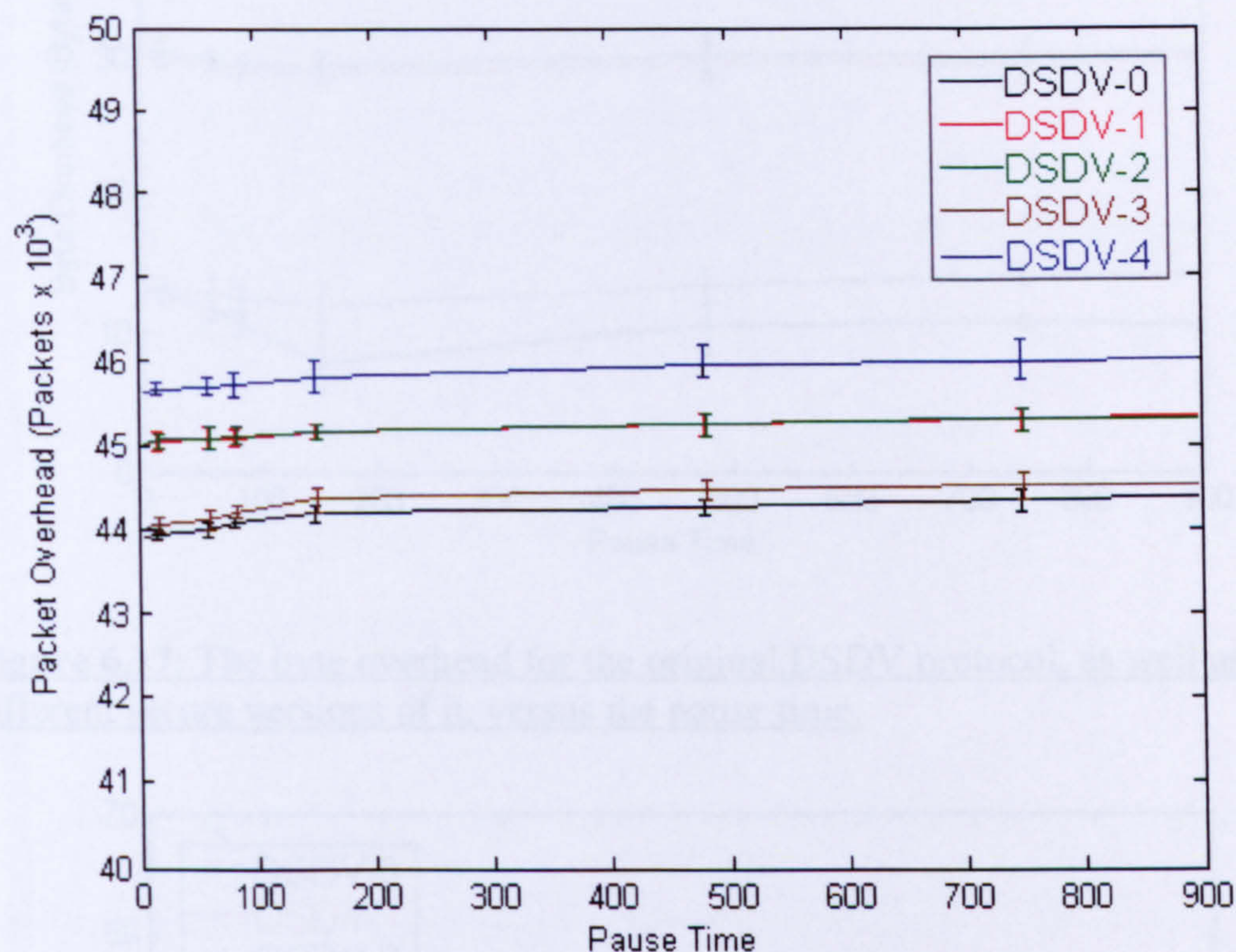


Figure 6.16: The packet overhead for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.

The figure above, as well as the one that follows, reflect the addition of extra data as part of securing DSDV. In the final metric, figure 6.18 depicts the median latency for the elapsed time between sending and receiving packets. Having experienced a higher value in both byte and packet overheads, we do expect the different secure versions of DSDV to exhibit a higher latency mainly due to network congestion, as the ranking for the overhead for each protocol presents.

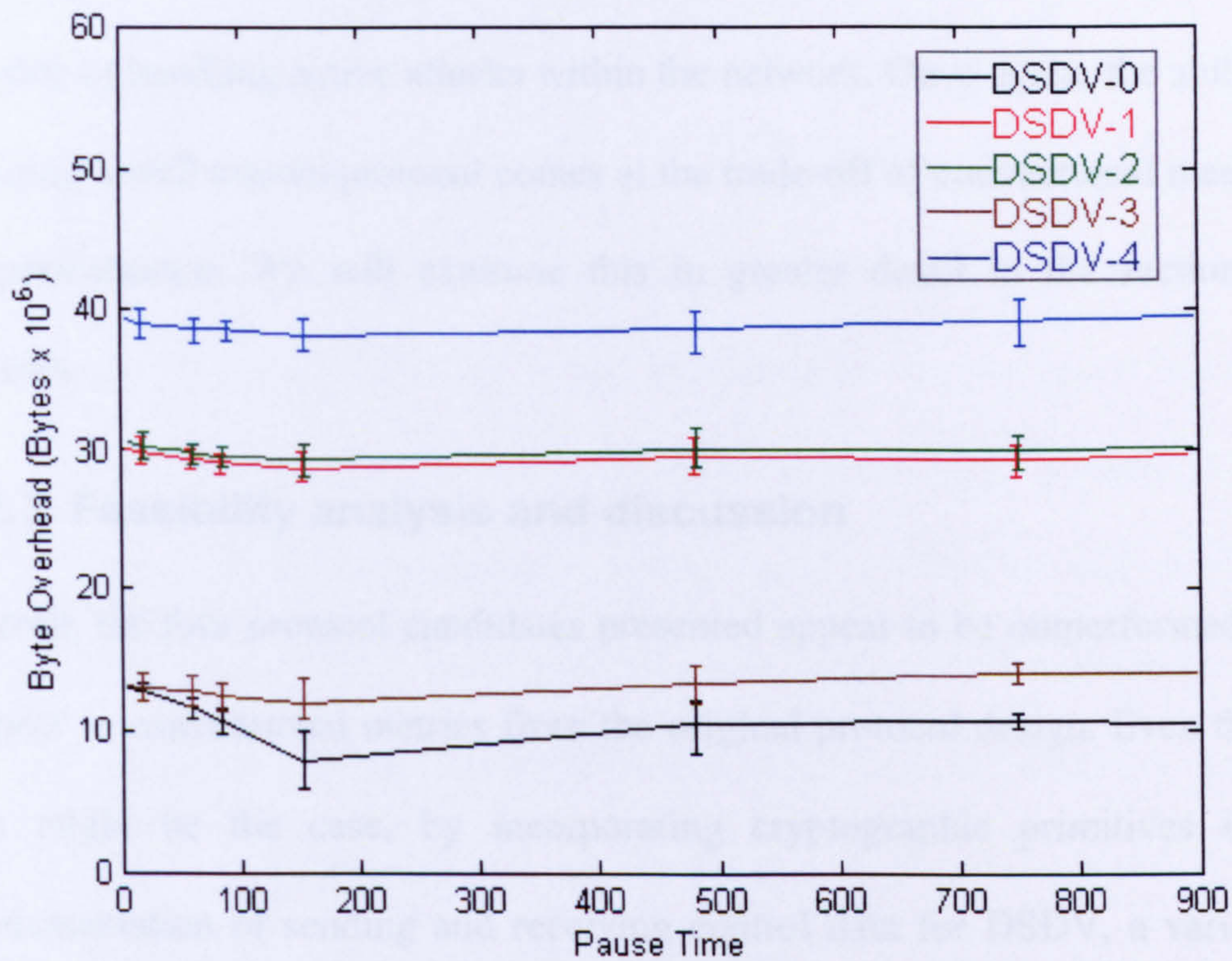


Figure 6.17: The byte overhead for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.

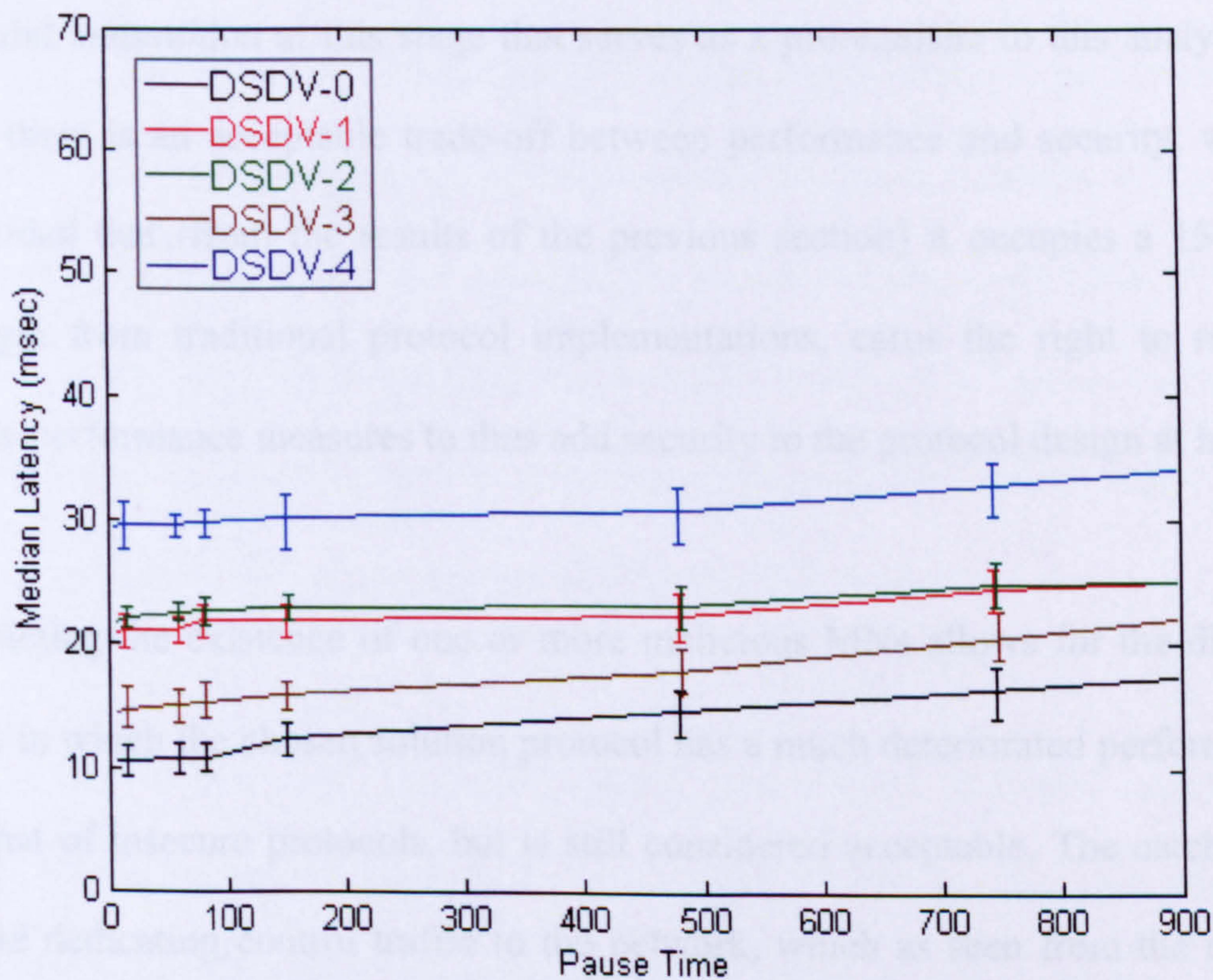


Figure 6.18: The median latency for the original DSDV protocol, as well as the 4 different secure versions of it, versus the pause time.

The above section summarises the alterations of the DSDV routing protocol capable of handling active attacks within the network. Once again, the ability of securing a well known protocol comes at the trade-off of conventional measures of performance. We will examine this in greater detail in the section that follows.

6.2.3 Feasibility analysis and discussion

Overall, the four protocol candidates presented appear to be outperformed with respect to conventional metrics from the original protocol design. Even though this might be the case, by incorporating cryptographic primitives in the implementation of sending and receiving control data for DSDV, a variety of active attacks could be prevented.

A valid assumption at this stage that serves as a prerequisite to this analysis is that there is an acceptable trade-off between performance and security, which provided that (from the results of the previous section) it occupies a 15-20% margin from traditional protocol implementations, earns the right to reduce static performance measures to thus add security to the protocol design at hand.

Assuming the existence of one or more malicious MNs allows for the distinct case in which the chosen solution protocol has a much deteriorated performance to that of insecure protocols, but is still considered acceptable. The catch is to avoid dedicating control traffic to the network, which as seen from the results presented is not the case.

As we can see, the packet delivery ratio is higher than that of the original protocol specification for all four secure implementations of DSDV. This is mainly due to the fact that as all four protocols provide the means for authenticating any incoming routing table update, fewer packet drops to destinations which are unknown occur. Furthermore, the four protocols can be split into a further two groups; the two which actively change the contents of the information being exchanged by encrypting it (DSDV-2 and DSDV-3) guarantee a higher delivery ratio, while the two which simply provide means of authenticating the addressee and/or the content (DSDV-1 and DSDV-4) perform less well.

This is justified as follows. Any malicious node with *maliciousLevel* above 00006 has the ability to generate its own hashes for any message that it forwards. Further to this and despite giving away its identity, a malicious node can alter any signed and hashed information that it obtains from the network, by altering them accordingly and attaching its own signature to the data. Without the knowledge of the system, this makes an adversarial node harder to detect on the network. Meanwhile, such information is not available when it is encrypted (either symmetrically or asymmetrically) to target a particular node.

In terms of byte and packet overhead, the results obtained match the performance expectation of a larger overhead due to the utilisation of cryptographic primitives. Commencing from DSDV-3, we can see a similar overhead to that of DSDV-0. This is because during asymmetric encryption, no further data is added to the information, apart from occasional padding to match

the length of the public key. Next in performance are DSDV-1 and DSDV-2; both these protocols transmit a hash of the message being exchanged as means of verifying the integrity of it. This justifies the virtually identical performance in terms of overhead that they both have. Finally, the worst performance comes from DSDV-4 which further to providing a hash for the message transmitted, also carries the signature of the originating node. As more data needs to be exchanged, the overhead increases by a further amount, compared to those previously depicted in the figures.

Also, as reflected in the median latency, there is a rise in latency at higher pause times due to the non-uniform distribution of node motion in the random waypoint mobility model. However, due to an increased overhead, secure protocols exhibit a higher latency than that of DSDV. This is because of higher congestion in the network while control data is being exchanged and also because the available network capacity is decreased by the increased overhead. A further contributing factor which in this case has not been considered is also the time taken to encrypt, sign and/or hash the data. Incorporating this into future simulations would yield a clearer difference between symmetric and asymmetric ciphers, as the latter of the two require more processing capacity and processing time to perform the enciphering and deciphering process than the former.

The feasibility of such a protocol, despite only changing one fundamental aspect in a well known design, unfolds well beyond the protocols already reviewed within this thesis. This is mainly due to the layout protocol architecture that we

have incorporated within Swarm, in which the task of deriving a secure design to meet a set of premeditated malicious standards, targets only the necessary elements of the system. Each MN utilises encryption to target control information towards particular nodes on the network. In a situation of high mobility, often a node finds itself completely unaware of its surrounding nodes that it can communicate with. Thus, despite achieving a better performance in terms of PDR, at the cost of a higher packet and byte overhead, realistically, any such implementation of DSDV such as the four seen in this section would cause a much lengthier time of convergence for routes on the network than that of the original protocol specification.

From the analysis and results presented in this section, what remains is to combine the protocol design characteristics derived for passive eavesdropping with the techniques seen within this section in an attempt to safeguard from both passive as well as active attacks. The section that follows addresses this.

6.3 Combining protocol characteristics with cryptography

Using the characteristics that assisted us in the specification of a protocol having as an objective to tackle the black hole problem, this section aims to incorporate, through Swarm, a number of cryptographic primitives that will help prevent active as well as passive attacks.

For this, we will examine the ways in which Protocol X (sub-section 6.1.2) dictates the exchange of routing control information on the network and attempt to extend its operation by incorporating cryptography within its operations. Again, for this we will use the technique described in the previous section, as it

was simulated upon DSDV. Namely, we will attempt to incorporate any of the classes contained within the secure package (sub-section 5.4.3) in Protocol X.

6.3.1 Network and scenario configurations

For the setup of this simulation we combine the two configurations already presented in sub-sections 6.1.1 and 6.2.1, allowing the *MaliciousLevel* of each node to vary so that to represent both passive, as well as active attacks. The number of nodes their properties, the transmission rate and the 2D space remain the same. There are still 10% of nodes considered malicious but their level of maliciousness (table 5.6) can vary from passive eavesdropping to active attacks, without taking into consideration any collusion of power. Thus each malicious MN is set to have a *MaliciousLevel* within the range of $[00003 - 00009]$, selected at random.

Building on our protocol, we will attempt to safeguard the performance of Protocol X in the presence of active as well as passive, non-colluding attackers.

In terms of security, we follow the assumptions of sub-section 6.2.1. Each node has a private and public key pair, with every node, having knowledge of every other node's public key. Any hash algorithms used (figure 3.8) are non-keyed and operate on the message or in this case packet alone. The signing of any message on the packet level, takes place using the node's private key, which is kept secret. Furthermore, the existence of a symmetric key between every node pair is also postulated.

In this case, from the four methods of communication that a node has on the packet level, namely, sending, receiving, dropping and forwarding packets, we allow Swarm to redefine three. Thence, for sending, receiving and forwarding control packets as specified by Protocol X, each node has the ability to apply any of the four security classes available. Thus, $(4 \times 3 \times 2) \times 3$ pairs surface, each giving different abilities to the receiving, transmitted and forwarded packet information, as described by the four classes within the secure package.

The success for each of the candidates was measured with respect to the level of intrusion detection achieved within the protocol. This, as defined in sub-section 5.4.3, allowed for 5 different levels of intrusion detection, with *not realising the occurrence of a malicious event* being the lowest and *isolating any information originating from a malicious node* being the highest.

6.3.2 Results obtained

The results obtained for Protocol X focused on the ways in which control information was relayed across the network. Once again, we selected to perform the simulation for 100 runs; each of which returned a secure alteration to the protocol. From this, the decision daemon in layer 1, returned the following preferences, as depicted in figure 6.19. This was derived for the two sending processes, namely sending and forwarding and reversed for the receiving process.

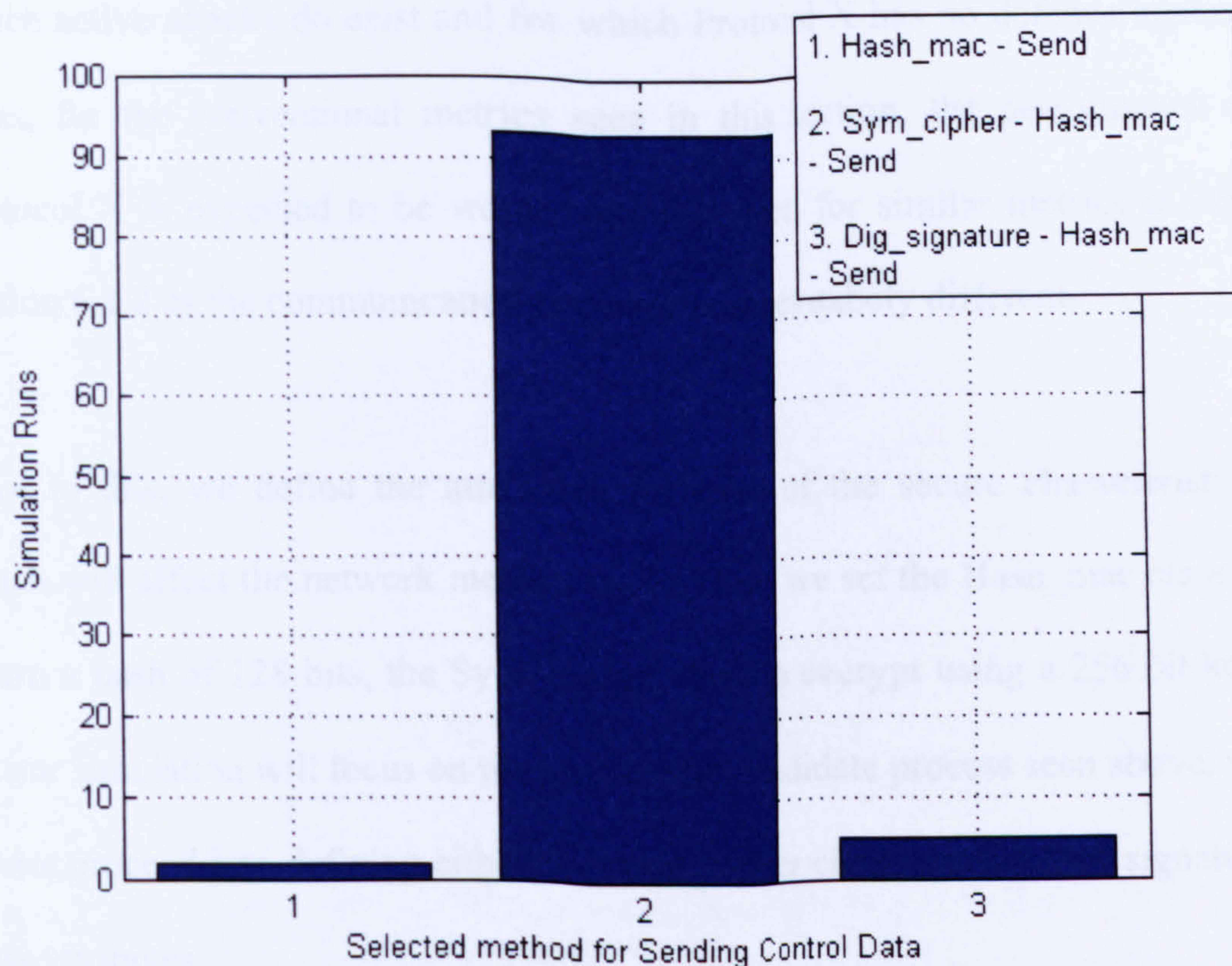


Figure 6.19: The three different protocol methods, as selected by Swarm, for sending, receiving and forwarding control packets within Protocol X.

From this, we can derive that unlike the results obtained for DSDV, there is a single candidate protocol based on Protocol X, which gives it the ability of tackling active attacks.

This is the process of symmetrically encrypting any message and then proceeding to obtaining and attaching the hash of that message to the data. Following this result, we advance by defining each of the two classes which in this case are required to further simulate the characteristics seen here within.

Our objective is to obtain a comparison with respect to the initial specification of Protocol X based on the four metrics defined in section 6.1.2. An aspect worth noting is that Protocol X is now being simulated in an environment in

which active attacks do exist and for which Protocol X has no defence against. Thus, for the conventional metrics seen in this section, the performance of Protocol X is expected to be worse than that seen for similar metrics in subsection 6.1.2 as the communication scenario is innervitably different.

Prior to this, we define the attributes for each of the secure characteristics, which will affect the network metrics used. Thus, we set the Hash_mac class to return a hash of 128 bits, the Sym_cipher class to encrypt using a 256 bit key. As our simulation will focus on the single best candidate process seen above, we do not proceed into defining either the Asm_cipher class, nor the Dig_signature class attributes.

The performance results presented are based on simulations over 50 runs with the Protocol X specification and the secure Protocol X specification, which we choose to label as Protocol Y. Once more, the error bars presented in each graph within the figures (as in section 6.1.2) represent the 95% confidence interval of the mean for each of the above metrics.

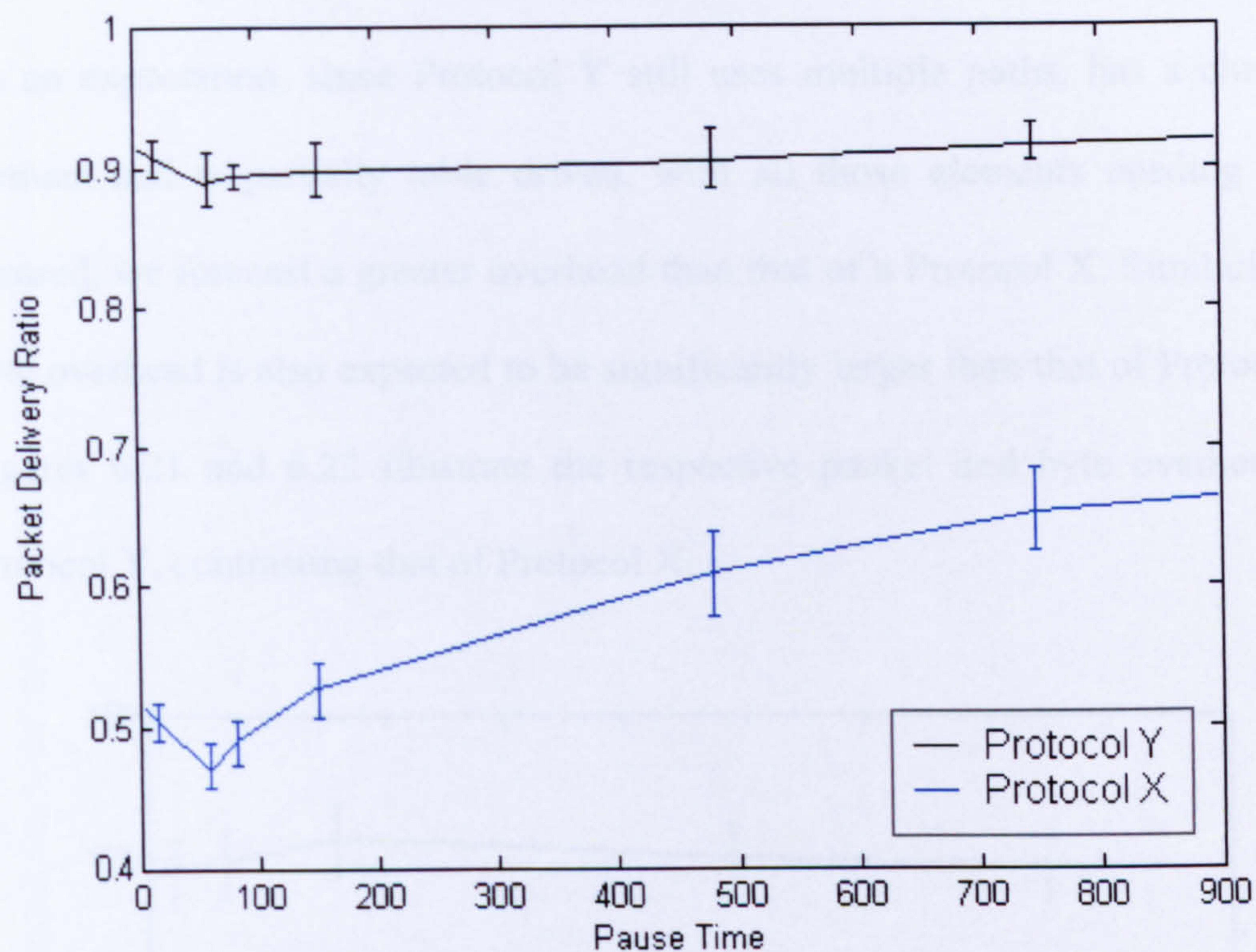


Figure 6.20: The packet delivery ratio for Protocol X and also Protocol Y versus the pause time.

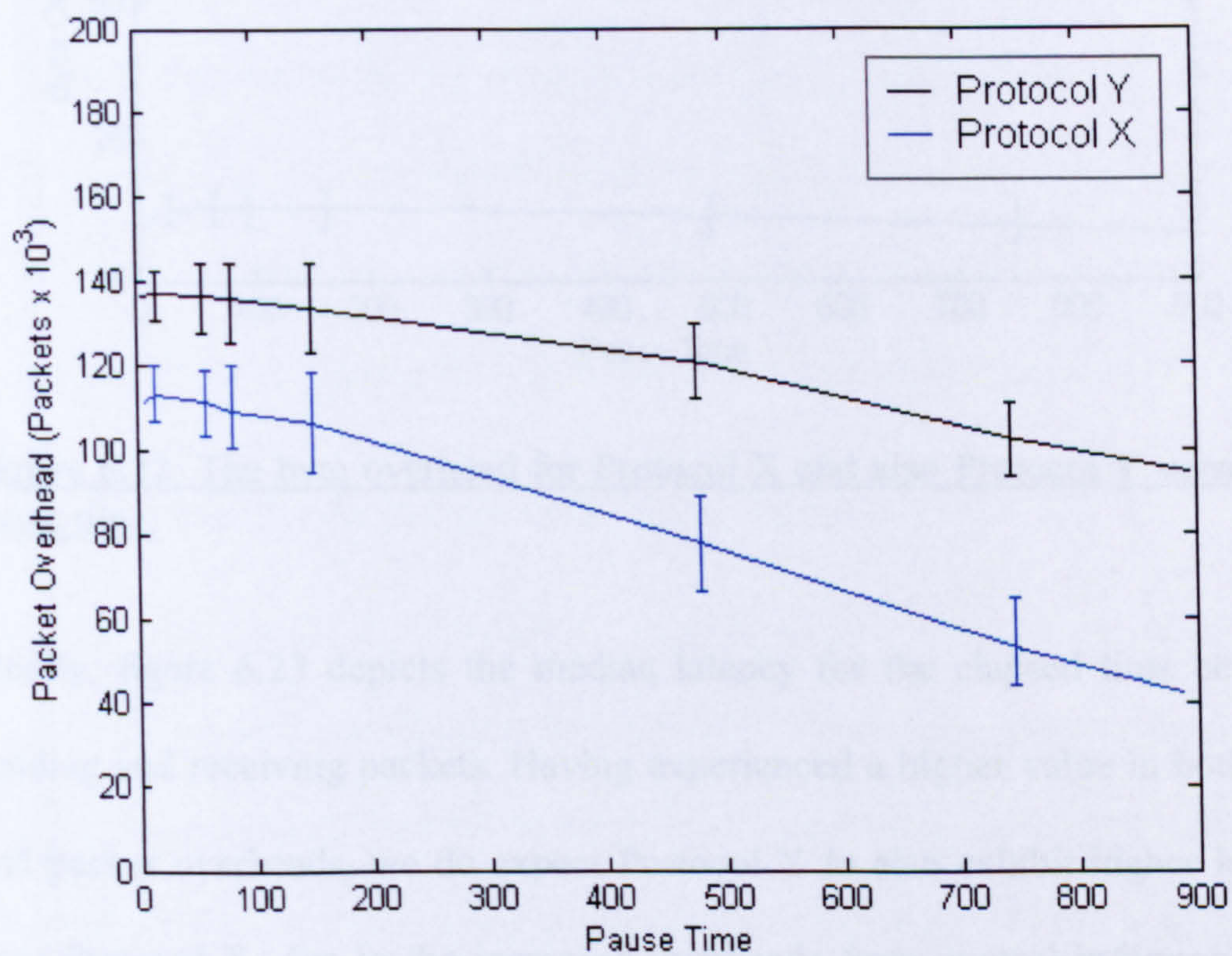


Figure 6.21: The packet overhead for Protocol X and also Protocol Y versus the pause time.

Figure 6.20 above details the packet overhead for Protocol X and Protocol Y. As an expectation, since Protocol Y still uses multiple paths, has a clustered element and is partially table driven, with all those elements needing to be secured, we forecast a greater overhead than that of a Protocol X. Similarly, the byte overhead is also expected to be significantly larger than that of Protocol X. Figures 6.21 and 6.22 illustrate the respective packet and byte overhead for Protocol Y, contrasting that of Protocol X.

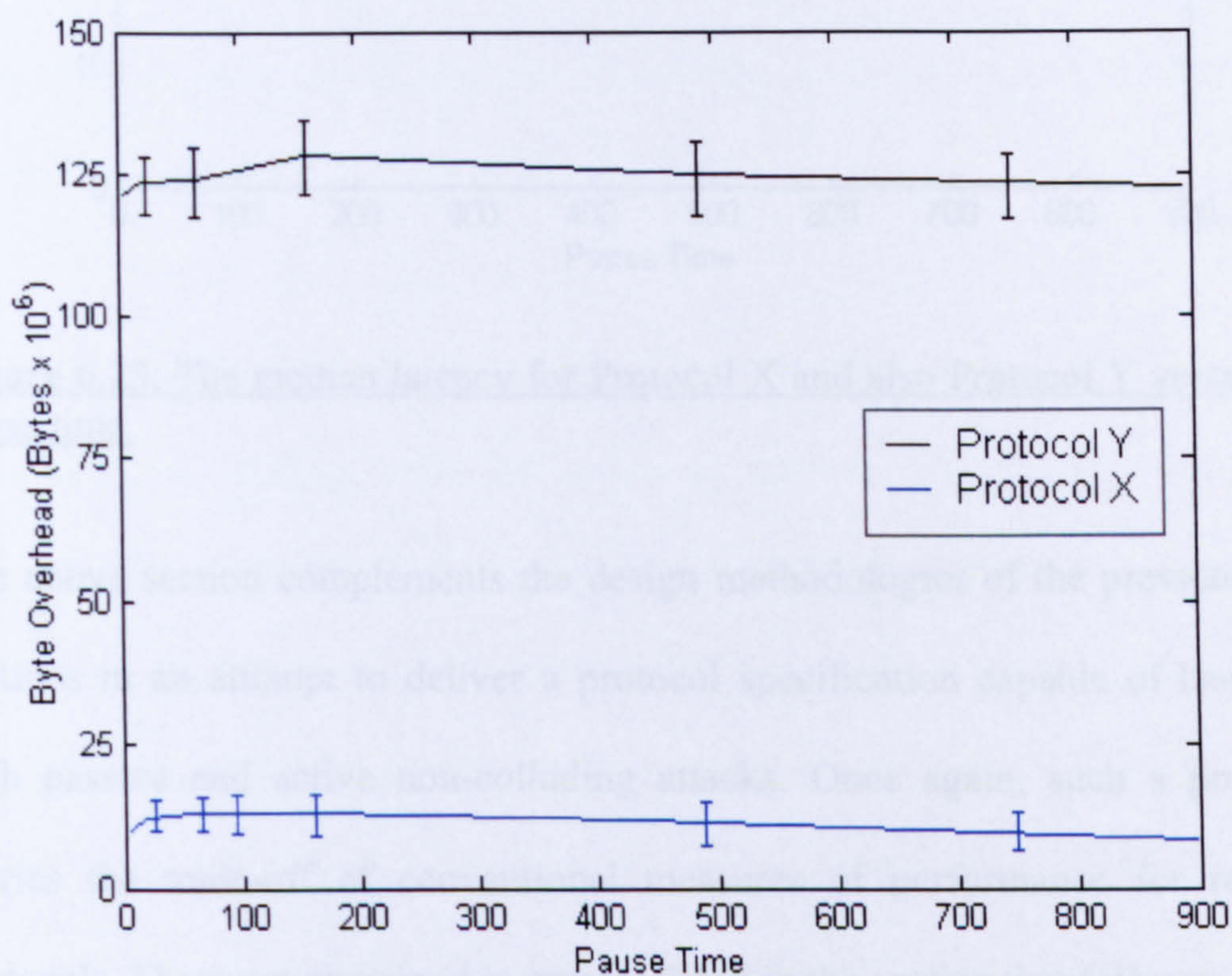


Figure 6.22: The byte overhead for Protocol X and also Protocol Y versus the pause time.

Finally, figure 6.23 depicts the median latency for the elapsed time between sending and receiving packets. Having experienced a higher value in both byte and packet overheads, we do expect Protocol Y to also exhibit higher latency than Protocol X, due to the increased overheads from control information, as specified through the secure sending, receiving and forwarding processes.

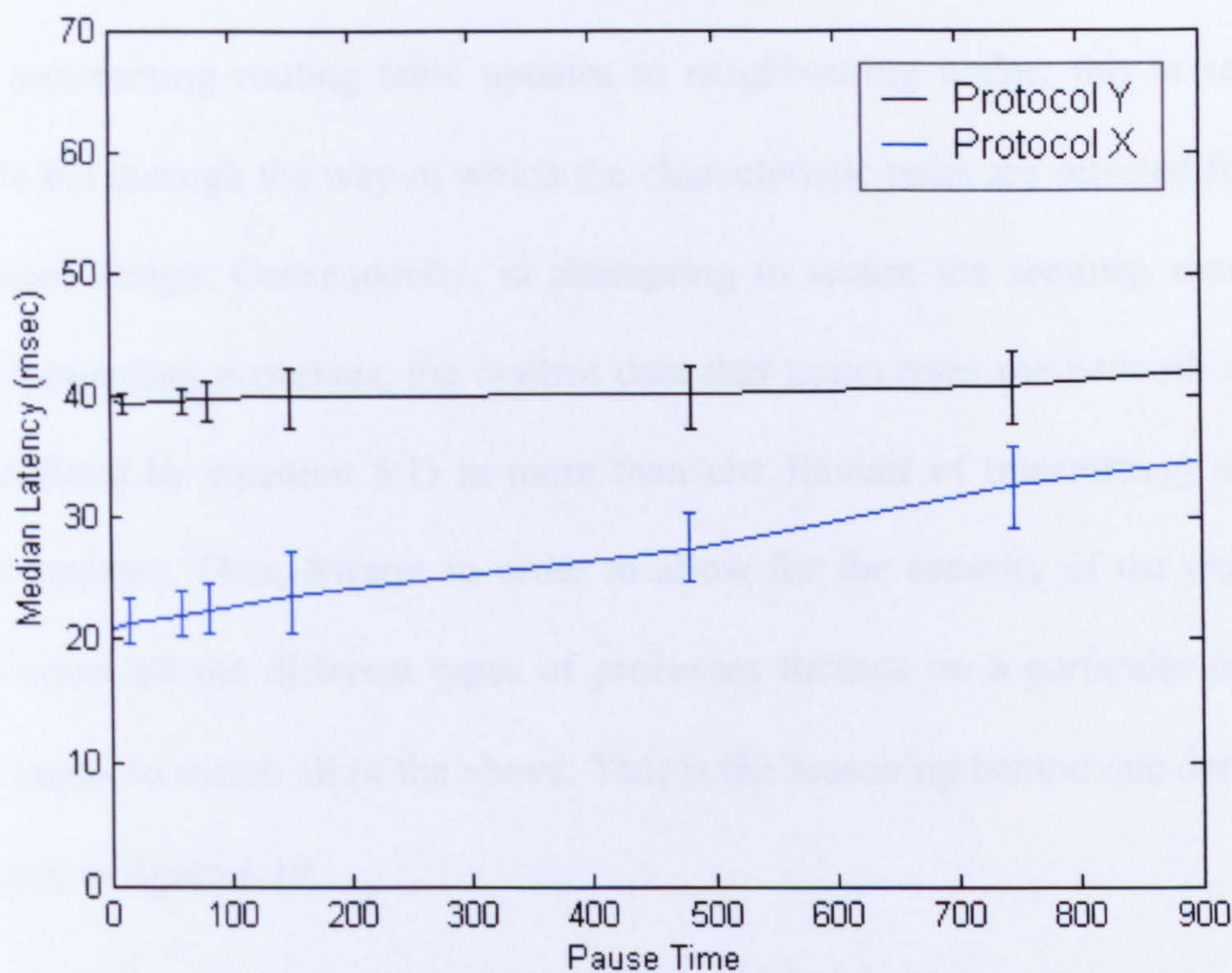


Figure 6.23: The median latency for Protocol X and also Protocol Y versus the pause time.

The above section complements the design methodologies of the previous two sections in an attempt to deliver a protocol specification capable of handling both passive and active non-colluding attacks. Once again, such a protocol carries the trade-off of conventional measures of performance for routing protocols. These are examined in greater detail in the section that follows.

6.3.3 Feasibility analysis and discussion

The design alterations that Swarm suggests in order to secure Protocol X focus on a single process, namely that of symmetrically encrypting any message and hashing and attaching that hash to it prior to transmission. Instead, in the case of securing DSDV four candidate protocols surfaced each of which had to be modelled separately.

Protocol X performs a number of additional tasks in terms of control data than just transmitting routing table updates to neighbouring nodes; this is seen in figure 6.8 through the way in which the characteristic pairs are selected for this protocol design. Consequently, in attempting to secure the sending, receiving and forwarding processes, the control data that transverses the network comes (as defined by equation 5.1) in more than one flavour of transmitting routing table updates. Thus, Swarm in order to allow for the security of the data and also cover all the different types of processes focuses on a particular process that seems to match all of the above. That is the reasoning behind one dominant process in figure 6.19.

From the metrics obtained, we can immediately notice that Protocol X is not in any way fit of handling active attacks on the network. The delivery ratio drops to approximately half of that observed in figure 6.9 where the simulation involved only passive attacks. As a result and since Protocol X is generating a variety of control data due to the seven characteristic pairs involved in its design, we can see that a fit protocol candidate for a particular type of an attack does not have the capacity of handling further malicious attacks that supersede its design.

In spite of this, it appears that the cryptographic primitives deployed for active attacks, also improve the performance from passive attacks. Again, when comparing figure 6.9 to figure 6.20, we can see that while for small pause times Protocol X is in the region of 0.85, Protocol Y improves on that metric at similar pause times by approximately 5 – 7 %.

In terms of byte and packet overhead, if we were to compare the results obtained in figures 6.21 and figures 6.22 to the byte and packet overheads generated for the secure DSDV protocol designs, we would notice an order of proportionality. Even though the packet overheads generated for the secure versions of DSDV vary from 44000 to 46000, we can see that the packet overhead for Protocol Y is approximately 30000 packets higher than that of Protocol X.

This is justifiable in the following way. For securing DSDV we focus on a single alteration in a way in which routing table updates are transmitted across the network. Reviewing the design of Protocol X (figure 6.8) we can see that it incorporates four hybrid characteristic pairs, each of which need to be secured against active attacks. Thus, the amount of control data and packets generated increased by a vast amount compared to DSDV. This is also reflected in the byte overhead (figure 6.22) where compared to that of the secure DSDV versions (figure 6.17) there is a difference of 100 Mbytes in the former and a maximum difference of 30 Mbytes in the latter. This creates a rough estimate for a figure of proportionality between the byte and packet overhead, with a difference in 1 packet as an overhead corresponding to a byte overhead of 250 bytes. As we have set each data packet to be 512 bytes in size, we can see that in order to secure against both passive and active attacks, approximately half the contents of a packet have to be loaded with control data in order to protect against both.

Again for the median latency, there is a rise in latency at higher pause times which is attributed to the non-uniform distribution of node motion in the random waypoint mobility model. Despite of the increase in the byte and packet overhead, the median latency does not follow a linear pattern of increase compared to those two. Thence, even though a higher congestion on the network while control data is being exchanged exists, as this information targets specific situations of communication defined by equation 5.1, an upper bound on the median latency experienced is present. In effect, the control data is only used when required and therefore contributes to a much lesser extend to the network congestion.

Protocol Y represents a design that has been build and improved to handle malicious behaviour across the network. Even though it achieves a much higher PDR at the cost of an increased byte and packet overhead, it improves on the median latency by illustrating that control traffic is only being used where necessary, with a maximum overhead of half the contents in a data packet. Again, the factors of processing capacity and time for utilising encryption have not being taken into account, nor has the issue of exchanging a unique symmetric key with every node been treated any differently than just a postulate. As much as these requirements might bring doubt to the design of this protocol, we must be prepared to utilise cryptographic primitives and also assume the existence of security associations between nodes (such as a symmetric key) if we are to handle attacks and disruptions in the unique routing capabilities of MANETs. Protocol Y delivers this capability.

6.4 Conclusions

This chapter has demonstrated the applicability of emergence within the design process of wireless routing protocols for MANETs. Prior to simulating a particular design we have put it to the test within the Swarm system. This has enabled us to further optimise each protocol and request that the system selects the characteristics that would be best capable of tackling the malicious communication scenario present within the network.

Despite of not having exhausted all the possible adversarial models, we have focused on the malignancy of an attack and looked at passive as well as active attacks. This divided the design process into three stages. In the first, we modelled through conventional characteristic pairs the ability of a protocol design to safeguard against black hole nodes. This process returned a number of values for each characteristic pair which allowed us to further simulate conventional metrics, such as the delivery ratio and the byte overhead.

By introducing cryptographic primitives and the secure protocol package within the protocol description of Swarm (sub-section 5.4.3) we went further in an attempt to secure the DSDV protocol specification. This lead to four candidate alterations in the transmission of routing table updates that were being exchanged. Once more, each one of the four implementations was simulated using conventional metrics so that to derive a performance comparison with respect to the original DSDV protocol.

As a final step, we attempted to apply the design of secure protocols upon the protocol candidate that was originally developed to deal with the presence of black holes on the network. In that process a dominant secure candidate was incorporated to the original design and a set of further results were obtained involving conventional metrics.

In each of the above cases, the security of a protocol always yielded a deterioration of performance with regards to the amount of information that was being exchanged between nodes. Inversely and again for all, a higher packet delivery ratio was obtained in the presence of the malicious communication scenario.

Finally, one of the most important aspects throughout this design process was the necessary justification of a particular design. The performance of each protocol was brought forward before an adaptive system capable of judging to a certain extent based on a number of parameters what the particular design should be. Thus, the way in which decisions were made in order to deliver a secure protocol have been presented and carefully put to the test by an adaptive event driven simulator. This has ultimately provided a further degree of freedom to the human element of the design process, giving us the chance to further focus on fundamental aspects of a protocol, while leaving issues specific to individual designs as parameters that can be modified and improved upon on the machine level.

Conclusions and future directions

7.1	Overview
7.2	The applicability of Protocol Y
7.3	Future directions
7.3.1	Improvements on the secure protocol package
7.3.2	A second generation pandemonium

This chapter summarises the research work presented. An overview of the framework of operation used for the development of secure protocols is presented, followed by an insight on the applicability of the one of the proposed protocol implementations, namely Protocol Y. Finally, work tangents for further development are presented, having as an objective the enhancement of the capabilities and usage of the system presented.

7.1 Overview

In this thesis we have addressed a host of issues pertaining to the design of secure wireless routing protocols for MANETs, using emergence. All issues, from existing protocol specifications to the design of secure implementations have been addressed. Although most of the performance analysis and simulation experiments have been performed considering non-colluding adversarial behaviour, a classification of malicious intent, as well as a short form of describing it (through a level definition) has been provided.

First we studied the nature of existing wireless routing protocols and provided a review of the available approaches to their categorisation. We then proceeded further into documenting the security aspects of communication within ad hoc networks; having done this we looked at ways and techniques that can help us prevent the occurrence of malicious behaviour within the network.

At that point, we proceeded into the design of an event driven simulation platform that can, with the assistance of emergence and adaptive modelling, select and implement secure protocols. Having labelled the software system developed as Swarm, we proceeded into the justification of particular characteristics that a protocol has based on an emergent selection process. This process, utilising adaptation is dependant on a number of worst case scenarios that can be experienced in a MANET, which represent the input of the Swarm system.

The output obtained focused on successful protocol candidates that have the ability to withstand by design particular types of adversarial behaviour. As a result, conventional metrics of performance, such as the delivery ratio, the overhead, as well as the average latency were calculated for standard simulation configurations. Further analysis and comments for each type of protocol selected, designed and simulated have been presented in the previous chapter.

This thesis gives a deeper insight into the problems that could be expected from next generations of secure wireless routing protocols. It is expected that future designs would be highly more versatile and complex. Such characteristics will entail a further focus on the mechanisms and processes that protocol architectures can obtain and also techniques for providing a justification in the selection and utilisation of particular attributes.

Instead of developing another formal specification for a selection process tied to particular parameters, we put together a system that can process resulting protocol architectures through emergence, in a way that allows for the mixing of particular attributes. From this, we used a selection process to match particular adversarial scenarios to candidate protocol architectures. As this process took place at the machine level, via the technique presented in this work, we had the ability to focus more easily on much more complex malicious behaviour and request from our adaptive system the fittest protocol description outside known solutions. Thus we have introduced a powerful design tool to carry forward research in MANET security.

7.2 The applicability of Protocol Y

From the resulting protocol architectures and by following a three-stage design process we developed a protocol specification capable of withstanding both passive as well as active attacks of non-colluding adversaries. The first step in this design process involved the identification of characteristic pair values in a way that would map the best possible candidate for withstanding passive attacks and the occurrence of black holes.

The specification of the candidate protocol developed by Swarm, which we labelled as Protocol X, was summarised in figure 6.8, offering a description that can allow further development. This was mainly due to the fact that in our definition of a protocol implementation (equation 5.1) specified the way in which different behaviours involving well defined characteristics could be obtained by applying that particular equation. As a result, the behaviour of Protocol X is well documented and can be summarised through equation 5.1 into 7 values within the range of $[0 - 10]$. In order to interpret these values, a further understanding of the way in which each one of the characteristic pairs is defined must be present. Thence, the results obtained, combined with the corresponding quantised range (figure 6.13) can be interpreted as a complete specification of Protocol X. The main reason for such a mechanical approach to the simulation of any protocol lies in the ability to input the specification into the Swarm system and obtain meaningful simulation results.

As Protocol Y builds on the specification of Protocol X, it puts to use a number of available cryptographic primitives, each one of which carries a pre-

communication assumption. This technique was first tested on the DSDV protocol delivering four candidates that had the ability to withstand active attacks. The main focus within each design was the update control data being transmitted by DSDV and the means of securing their exchange on the network. Simulation results once more confirmed the trade-off of security with respect to conventional metrics, exchanging higher overheads of network traffic for a higher delivery ratio of data. In the second step of the design process, a known protocol was offered a number of security extensions detailing the control data being generated for DSDV to function coherently.

The third step of the protocol design aimed to secure the derived implementation of Protocol X, responsible for handling passive attacks, so that it could also handle active attacks as well. Protocol Y based its operation on a fundamental assumption; a symmetric key between each node-pair within the network. Even though this can be challenged, it represents an add-on in the list of prerequisites.

Aspects such as a fully charged set of batteries on the device come to be added to the secure exchange of symmetric keys prior to deployment of the MNs in a highly mobile environment. Such is the nature of ad hoc networking to assume that a minimum number of prerequisites are present prior to any communication taking place. Adding a security requirement that relates to layers other than the physical layer should come as no surprise; it simply represents the need to also take into account factors related to those layers. After all, the physical layer

merely enables communications; layers above that are responsible for the exchange and the security of the information transmitted.

From a design perspective, having tackled the last out of the five digits that describe the *maliciousLevel* of a MN, the next step would be to feed Protocol Y in a number of colluding adversarial scenarios. In this attempt, a specification (from the already utilised elements of the secure protocol package) would be obtained that would have the ability to match more complicated adversarial behaviour.

7.3 Future directions

From this effort, two directions for further work, one focusing on emergence and one on wireless routing, are immediately apparent. For emergence, we consider the development of a second generation pandemonium that can avoid loopholes in the adaptive design phase. For wireless routing, the ways in which extensions to the secure protocol package specification can be designed are contemplated.

7.3.1 Improvements on the secure protocol package

In describing the general characteristics of a wireless routing protocol, we selected seven characteristic pairs that were treated equally within our solution environment. Even though this was the case, a number of further characteristics of smaller importance can also be taken into consideration, involving packet types, reply requests and so on. As these would not be as general as those used in this work, a different weighting would have to be given to each one, perhaps as a subset of one of the seven general characteristics seen above.

Furthermore, a number of more advanced cryptographic primitives can be developed for usage within the secure protocol package. Such techniques could include threshold cryptography (sub-section 3.5.2.1) as well as protocol correctness techniques that would help avoid conflicts within a design.

A potential hazard in such an expansion relates to the complexity of the system. Thus, caution should be taken with regards to such additions, as they would go beyond affecting the sending, receiving and forwarding methods available through a protocol specification to each node. Inversely, any methods specified would have to hold greater generality than the ones used, so that to incorporate actions such as the transmission of packets but also facilitate techniques such as threshold cryptography. This would result in a revisit of methods such as *send* for transmitting packets, with methods like *useSecureSymmetricChannel* for the transmission of symmetrically encrypted control data. In such a system, correctness proofs could be built into the system as a verification technique for the proposed protocol architecture.

7.3.2 A second generation pandemonium

In the adaptive system developed, little control is given to the user with regards to the selection and biasing of the adaptive properties involved in the selection process. Thus, the definition of our adaptive plan facilitates for improvement within neighbouring structures, upon finding a partially successful candidate.

Our motivation behind a second generation system lies that designs presented in the current version of Swarm can lead to local maxima appearing as the best possible solutions within a much larger environment. For the purpose of our

work, this fact was outweighed from the numerous repetitions and random initial conditions, which gave us the opportunity to select characteristic descriptions.

In a second generation simulator, we suggest a much more complex adaptive plan, having the capability to overcome this limitation, by not only improving on local values, but also randomly selecting other regions within the environment. Consequently, a second generation pandemonium would incorporate a technique for further reporting on any emergent properties discovered and allow for a linear, as well as a non-linear decision making process in the adaptive plan. A way of achieving this would be to introduce an “evil daemon” in layer 1 that instead of selecting optimal values, would review worst case scenarios as best and vice versa. Naturally, such a process would have to take place for a limited amount of time only, for a successful protocol to emerge from the system.

List of publications

2005	
	I. Pavlosoglou , M. S. Leeson, R. J. Green, "Applying emergence to the design of routing protocols for the security of wireless ad hoc networks", SecureComm 2005, First IEEE/CreateNet International Conference on Security and Privacy for Emerging Areas in Communication Networks , September 2005
2004	
	R. Rejeb, I. Pavlosoglou , M. S. Leeson and R. J. Green, "Management Issues in Transparent Optical Networks", 6th International Conference on Transparent Optical Networks (ICTON 2004), vol. 1, pp. 248-254, Wroclaw, July 2004.
2003	
	I. Pavlosoglou , T Stergiou, M S Leeson and R J Green, "A proposed scheme for securing IEEE 802.11 wireless LANs", Proceedings of the London Communications Symposium 2003, pp. 265-268, Sep 2003.
	I. Pavlosoglou , M S Leeson and R J Green, "Spotting emergence in wireless routing protocols", Proceedings of the London Communications Symposium 2003, pp. 269-272, Sep 2003.
	R. Rejeb, I. Pavlosoglou , M. S. Leeson and R. J. Green, "Securing All-Optical networks", Proceedings of the 5th International Conference on Transparent Optical Networks (ICTON 2003) , vol. 1, pp. 87-90, Jul 2003.
	I. Pavlosoglou , M. S. Leeson and R. J. Green, "Towards bottom-up network architectures", Proceedings of the 4th annual postgraduate symposium PGNet 2003, pp. 21-24, Jun 2003.
2002	
	I. Pavlosoglou , R. Ramirez-Iniguez, M. S. Leeson, and R. J. Green, "A Security Application of the Warwick Optical Antenna in Wireless Local and Personal Area Networks", Proceedings of the London Communications Symposium 2002, pp. 225-228, Sep 2002.
	I. Pavlosoglou , "The need for securing wireless communications", IEE Write around the world competition, runner-up prize (2 nd and 3 rd place), 2002.

References

- [1] M. El-Sayed, J. Jaffe, "A view of telecommunications network evolution", IEEE Communications Magazine, vol. 40, no. 12, pp. 74-81, 2002
- [2] B. O'Hara, A. Petrick, "The IEEE 802.11 Handbook: A Designer's Companion", Standards Information Network IEEE Press, 1999
- [3] Various, "Specification of the Bluetooth System", Bluetooth SIG, http://www.bluetooth.com/pdf/Bluetooth_11_Specifications_Book.pdf, 1999, pp. 19-23
- [4] Various, "Technical Summary of IrDA DATA and IrDA CONTROL", Infrared Data Association, <http://www.irda.org/standards/standards.asp>, 2000
- [5] R. Caceres, L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments", IEEE Journal of Selected Areas in Communications, vol. 13, no. 5, 1995, pp. 850-857
- [6] H. Deng, W. Li D. P. Agrawal, "Routing security in wireless ad hoc networks" IEEE Communications Magazine, vol. 40, no. 10, 2002, pp. 70 – 75
- [7] C-K Toh, "Wireless ATM and ad-hoc networks, protocols and architectures", Kluwer Academic Publishers, London, 1997, pp. 193 – 194
- [8] P. Yalagandula, "A Survey on Security Issues in Wireless Networks", University of Texas, Department of Computer Science, http://www.cs.utexas.edu/users/ypraveen/surveys/wlan_security, 2000
- [9] V. Karpijoki, "Security in Ad Hoc Networks", Helsinki University of Technology, Telecommunications and Software Laboratory, http://www.hut.fi/~vkarpijo/netsec00/netsec00_manet_sec.ps, 2000
- [10] L. Zhou, Z. J. Haas, "Securing Ad Hoc Networks", IEEE Networks Magazine, vol. 13, no. 6, 1999, pp. 24 – 30
- [11] M. Gast, "802.11 Wireless Networks: The Definitive Guide", O'Reilly, 2002, pp. 267-269
- [12] R. Prasad, editor, "Universal wireless personal communications", Artech House, Norwood, MA, 1998
- [13] S. Sampei, editor, "Applications of Digital Wireless Technologies to Global Wireless Communications", Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997
- [14] P. W. Anderson, "Emergence", Science Board Meeting 2003, Santa Fe Institute, Santa Fe, New Mexico, USA, <http://www.santafe.edu/files/gems/scienceboard03/PhilipAnderson.pdf>, 2003
- [15] D. M. Gordon, "Ants at Work: how an insect society is organized", Free Press, Simon & Schuster, 2000
- [16] D. Ince, "Developing distributed and e-commerce applications" Addison-Wesley, 2002, pp. 575 - 613
- [17] S. Wolfram, "A new kind of science", Wolfram media Inc., 2002

- [18] J.-L. Deneubourg, S. Aron, S. Goss, J.M. Pasteels, "The self-organizing exploratory pattern of the Argentine ant", *Journal of Insect Behavior*, vol. 3, 1990, pp. 159 - 168
- [19] A. Ilachinski, "Cellular Automata: A Discrete Universe", World Scientific, 2001, pp. 63 – 71
- [20] K. Koffka, "Principles of Gestalt Psychology", New York: Harcourt-Brace, 1935, pp. 175 – 176
- [21] H. Özbay, "Introduction to feedback control theory", London CRC Press, 2000, pp. 52 – 104
- [22] S. Barnett, R.G. Cameron, "Introduction to mathematical control theory", 2nd edition, Oxford Clarendon Press, 1985, pp. 74 – 98
- [23] G. Di Caro, M. Dorigo, "AntNet: Distributed Stigmergetic Control for Communications Networks", *Journal of Artificial Intelligence Research*, vol. 9, 1998, pp. 317 – 365
- [24] G. Di Caro, M. Dorigo, "AntNet: a mobile agents approach to adaptive routing", Technical Report IRIDIA/97-12, Universit Libre de Bruxelles, Belgium, 1997
- [25] G. Di Caro, M. Dorigo, "Mobile agents for adaptive routing", *Proc. 31st Hawaii International Conference on System Sciences*, IEEE Computer Society Press, Los Alamitos, CA, 1998, pp. 74-83
- [26] R. E. Kahn, S. Gronemeyer, J. Burchfiel, R. Kunzelman, "Advances in Packet Radio Technology" *Proceedings of the IEEE*, vol. 66, no. 11, 1978, pp. 1468 – 1496
- [27] J. Jubin, J. Tornow, "The DARPA Packet Radio Network Protocols", *Proceedings of the IEEE*, vol. 75, no. 1, 1987, pp. 21 – 32
- [28] E. M. Royer, C-K. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", *IEEE Personal Communications Magazine*, vol. 6, no. 2, 1999, pp. 46 – 55
- [29] Institution of Electrical and Electronic Engineers (IEEE), "Wireless LAN Medium Access Control (MAC) and Physical layer (PHY) specification", LAN MAN Standards of the IEEE Computer Society, IEEE Standard 802.11, 1997
- [30] X. Zou, B. Ramamurthy, S. Magliveras, "Routing Techniques for Wireless Ad Hoc Networks – Classification and Comparison", *Proceedings of the Sixth World Multi-conference on Systemics, Cybernetics, and Informatics – SCI*, July 2002
- [31] K. Inkinen, "New Secure Routing in Ad Hoc Networks: Study and Evaluation of Proposed Schemes", Technical Report, Laboratory of Multimedia, Helsinki University of Technology, 2004
- [32] J. Broch, D. A. Maltz, D. B. Johnson, Y-C Hu, J. Jetcheva, "A performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols", *Proc. of the 4th ACM International Conference on Mobile Computing and Networking (MobiCom'98)*, 1998, pp. 25 – 30
- [33] D. B. Johnson, D. A. Maltz, "Dynamic source routing in ad hoc wireless networks", *Mobile Computing*, ed. T. Imielinski, H. Korth, Kluwer Academic Publishers, 1996, pp. 153 – 181
- [34] D. B. Johnson, D. A. Maltz, "Protocols for adaptive wireless and mobile computing", *IEEE Personal Communications*, vol. 3, no. 1, 1996, pp. 34 – 42

- [35] D. B. Johnson, D. A. Maltz, J. Broch. "DSR The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks. In *Ad Hoc Networking*", ed. C. E. Perkins, Addison-Wesley, 2001, pp. 139 – 172
- [36] C. E. Perkins, P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers", In *Proceedings of the SIGCOMM '94 Conference on Communications Architecture, Protocols, and Applications*, 1994, pp. 234 – 244
- [37] M. Steenstrup (editor), "Routing in Communication Networks", Englewood Cliffs, London, Prentice Hall, 1995, pp. 84 – 87
- [38] L. R. Ford Jr, D. R. Fulkerson, "Flows in Networks", Princeton, NJ, Princeton University Press, 1962
- [39] D. P. Bertsekas, R. G. Gallager, "Data Networks", Englewood Cliffs, NJ: Prentice-Hall, 1987
- [40] Y. Lu, W. Wang, Y. Zhong, and B. Bhargava, "Study of distance vector routing protocols for mobile ad hoc networks," *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom 2003)*, 2003, pp. 187 – 194
- [41] C. E. Perkins, E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing", *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999, pp. 90 – 100
- [42] E. Royer and C. Perkins. Multicast operation of the ad-hoc on-demand distance vector routing protocol. *Proceedings of the ACM Mobicom'99*, August 1999, pp. 207 – 218
- [43] C. Perkins. "Ad-Hoc On-Demand Distance Vector Routing", *MILCOM'97 panel on Ad-Hoc Networks*, Monterey, CA, November 3, 1997
- [44] C. Perkins, E. Royer, S. Das, "Ad hoc on-demand distance vector routing", IETF MANET Working Group, IETF Internet Draft, draft-ietf-manet-aodv12.txt, work in progress, November 2002
- [45] V. D. Park, M. S. Corson, "Temporary Ordered Routing Algorithm (TORA) Version I: Functional specification", IETF MANET Working Group, IETF Internet Draft, draft-ietf-manet-tora-spec-01.txt, work in progress, August 1998
- [46] V. D. Park, M. S. Corson, "A performance comparison of the Temporally Ordered Routing Algorithm (TORA) and Ideal Link-state routing", In *proceedings of the IEEE Symposium on Computers and Communication'98*, 1998, pp. 120 – 129
- [47] V. D. Park, M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", *Proceedings of INFOCOM'97*, 1997, pp. 1405 – 1413
- [48] M. S. Corson and V. D. Park, "An Internet MANET Encapsulation Protocol (IMEP) Specification", IETF MANET Working Group, IETF Internet Draft, draft-ietf-manetimep-spec-01.txt, work in progress, November 1997
- [49] J. P. Macker and M. S. Corson, "Mobile ad hoc networking and the IETF", *Mobile Computing and Communications Review*, vol. 2, no. 1, 1998, pp. 9 – 14

- [50] Z. J. Haas, "A new routing protocol for the reconfigurable wireless networks", In Proc. of IEEE 6th International Conference on Universal Personal Communications (ICUPC'97), San Diego, California, USA, 1997, pp. 562 – 566
- [51] Z. J. Haas, M. R. Pearlman, "The Performance of Query Control Schemes for the Zone Routing Protocol" In Proceedings of ACM SIGCOMM'98 conference, September 1998, pp. 167 – 177
- [52] Z. J. Haas, M. R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", IETF MANET Working Group, IETF Internet Draft, draft-ietf-manet-zrp-01.txt, work in progress, August 1998
- [53] C.-C. Chiang, H-K Wu, W. Liu, M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks", The IEEE Singapore International Conference on Networks, 1997, pp. 197 – 211
- [54] C. Chiang, M. Gerla, and L. Zhang. "Shared tree wireless network multicast", In Proc. of 6th Int. Conf. on Computer Communications and Networks, September 1997, pp. 28 – 33
- [55] G. Di Caro and M. Dorigo. "AntNet: Distributed Stigmergetic Control for Communications Networks", Journal of Artificial Intelligence Research, vol. 9, 1998, pp. 317 – 365
- [56] G. Di Caro and M. Dorigo, "Mobile agents for adaptive routing", In Proc. of the 31st Hawaii International Conference on System Sciences, IEEE Computer Society Press, Los Alamitos, CA, USA, 1998, pp. 74 – 83
- [57] M. den Besten, T. Stutzle, and M. Dorigo. "Ant colony optimization for the total weighted tardiness problem", In Proc. of the Parallel Problem Solving from Nature Conference, 2000.
- [58] G. Di Caro and M. Dorigo. "An adaptive multi-agent routing algorithm inspired by ants behaviour", In Proceedings of PART98 - 5th Annual Australasian Conference on Parallel and Real-Time Systems, 1998, pp. 261 – 272
- [59] G. Di Caro and M. Dorigo. "Ant colonies for adaptive routing in packet-switched communications networks". In A. E. Eiben, T. Back, M. Schoenauer, and H.-P. Schwefel, editors, Proceedings of PPSN-V, Fifth International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 1998, pp. 673 – 682
- [60] V.P. Volny, D.M. Gordon, "Genetic basis for queen-worker dimorphism in a social insect", Proc. Natl. Acad. Sci. USA, vol. 99, no. 6108-6111, <http://ant.stanford.edu/volnyandgordon2002.pdf>, 2002
- [61] M. Dorigo, V. Maniezzo, A. Colomi, "The Ant System: Optimization by a colony of cooperating agents" IEEE Trans. Syst, Man, Cybernetics, vol. 26, no.2, 1996, pp. 29 – 41
- [62] M. Dorigo, L. M. Gambardella, "Ant colony System: A Cooperative Learning Approach to the Travelling Salesman Problem", IEEE Trans. On Evolutionary Computation, vol. 1, no. 1, 1997, pp. 53 – 66
- [63] M. Dorigo, V. Maniezzo, A. Colomi, "Distributed Optimization by Ant Colonies", Proceedings of ECAL91 – European Conference on Artificial Life, Elsevier Publishing, 1991, pp 134 – 142

- [64] M. Dorigo, V. Maniezzo, A. Colomi, "An Investigation of some properties of an Ant Algorithm", Proceedings of the Parallel Problem Solving From Nature Conference (PPSN92), Brussels, Belgium, Elsevier Publishing, 1992, 509 – 520
- [65] C. Bierwith, "A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms", Department of Economics, University of Bremen, Germany, OR SPEKTRUM, vol. 17, no. 2-3, 1995, pp 87 – 92
- [66] G. Di Caro, M. Dorigo, Mobile Agents for Adaptive Routing, Technical Report, IRIDIA/97-12, Universit Libre de Bruxelles, Belgium, 1997
- [67] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Ant-based Load Balancing in Telecommunications Networks", Adaptive Behavior, Vol. 5, No. 2, 1996, pp. 169 - 207
- [68] M. Gunes, U. Sorges, I. Bouaziz, "ARA – The Ant-Colony Based Routing Algorithm for MANETs", In International Conference on Parallel Processing Workshops (ICPPW'02), Vancouver, Canada, 2001
- [69] M. Roth, S. Wicker, "Termite: Emergent Ad-Hoc Networking", The Second Mediterranean Workshop on Ad-Hoc Networks, Medhia, Tunisia, 2003
- [70] M. S. Corson, A. Ephremides, "A distributed Routing Algorithm for Mobile Wireless Networks", ACM / Baltzer Wireless Networks Journal, vol. 1, no. 1, 1995, pp 61 – 81
- [71] C-K Toh, "Associativity-Based Routing for Ad-Hoc Mobile Networks", IEEE Wireless Personal Communications, vol. 4, no. 2, 1997, pp. 1 - 36
- [72] G. Brassard, P. Bratley, "Fundamentals of Algorithmics", Prentice-Hall, 1996, pp. 79 – 85
- [73] I. A. Wagner, M. Lindenbaum, A. M. Bruckstein, "ANTS: Agents, networks, trees and subgraphs", Future Generation Computer Systems, vol. 16 no. 8, 2000, pp. 915 – 926
- [74] I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein, "Efficient graph search by a smell-oriented vertex process. Annals of Mathematics and Artificial Intelligence, no. 24, 1998, pp. 211 – 223
- [75] S. Harris (editor), "The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force", IETF Internet draft, <http://www.ietf.org/tao.html>, August 2001
- [76] IETF MANET Charter, <http://www.ietf.org/html.charters/manet-charter.html>, 1998
- [77] S. Corson, J. Macker, "Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations", IETF MANET Working Group, IETF Internet Draft, rfc2501.txt, January 1999, pp. 7 – 8
- [78] R. Sanhu, "Access Control: The Neglected Frontier", In Proceedings of the First Australian Conference on Information Security and Privacy, Wollonlong, Australia, June 23 – 26, 1996
- [79] B. Schneier, "Secrets and Lies", John Willey and Sons Inc., 2000
- [80] B. Schneier, "Beyond Fear", Copernicus Books, 2003, pp. 103 – 117
- [81] V. Lenin, "The chain is no stronger than its weakest link" Pravda no 67, May 27 (June 9) 1917

- [82] J. William, "The varieties of religious experience", Harvard University Press, 1985, pp. 153 – 154
- [83] M. Wilkinson, J. L. Thorley, P. Upchurch, "A chain is no stronger than its weakest link: Double decay analysis of phylogenetic hypotheses", *System Biology Journal*, vol. 49, no. 4, 2000, pp. 754 – 776
- [84] J. Vitek, G. Castagna, "Towards a Calculus of Secure Mobile Computations", In *Workshop on Internet Programming Languages*, Chicago, Ill., May 1998. reprinted in *Electronic Business Objects*, Ed. Tsichritzis, University of Geneva, 1998, pp. 31 - 46
- [85] M. Graff, K. R. van Wyk, "Secure coding : principles and practices", O'Reilly (Cambridge), 2003
- [86] J. Viega, G. McGraw, "Building secure software : how to avoid security problems the right way", Addison-Wesley (London), 2002
- [87] T. Lauschner, A. Macedo, S. Campos, "Formal verification and analysis of a routing protocol for ad hoc networks" July 1997, <http://jurua.dcc.fua.br/tanara/artigo.html>
- [88] K. Bhargavan, C. A. Gunter, D. Obradovic, "Formal verification of standards for distance vector routing protocols" *Journal of the ACM*, vol. 48, no. 4, 2002, pp. 538 – 576
- [89] I. Zakiuddin, M. Goldsmith, P. Whittaker, P. H. B. Gardiner, "A methodology for model-checking ad-hoc networks" *Lecture Notes in Computer Science*, vol. 2648/2003, no. 0302-9743, January 2003, pp. 181 – 196
- [90] J. Edney, B. Arbaugh, "Real 802.11 Security: Wi-Fi Protected Access and 802.11i", Addison Wesley, 2003, pp. 7 – 21
- [91] T. Akin, "Hardening Cisco Routers", O'Reilly, 2002, pp. 3 – 4
- [92] B. Awerbuch, D. Holmer, H. Rubens, "Provably Secure Competitive Routing against Proactive Byzantine Adversaries Via Reinforcement Learning", Technical Report version 2, Department of Computer Science, Johns Hopkins University, Baltimore, MD, 2003, <http://citeseer.ist.psu.edu/666989.html>
- [93] K. Inkinen, "New Secure Routing in Ad Hoc Networks: Study and Evaluation of Proposed Schemes", Helsinki University of Technology, Laboratory of Multimedia, 2004, <http://citeseer.ist.psu.edu/716222.html>
- [94] Y.-C. Hu, A. Perrig and D. B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks", *Proc. of the Eighth Annual International Conference on Mobile Computing and Networking (MobiCom 2002)*, 2002, pp. 12 – 13
- [95] P. Michiardi and R. Molva, "Ad hoc network security," *ST Journal of System Research*, Volume 4, N1, March 2003 pp. 11 – 15
- [96] M. Corner and B. Noble, "Protecting applications with transient authentication", In *First ACM/USENIX International Conference on Mobile Systems, Applications and Services (MobiSys'03)*, San Francisco, CA, May 2003
- [97] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Wormhole detection in wireless ad hoc networks", Department of Computer Science, Rice University, Tech. Rep. TR01-384, June 2002

- [98] J. Mitchell, M. Mitchell, U. Stern, "Automated analysis of cryptographic protocols using MurOE", In Proceedings of the 1997 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, May 1997, pp. 141 – 151
- [99] C. Meadows, "Applying formal methods to the analysis of a key management protocol", Journal of Computer Security, vol. 1 no. 1, 1992, pp. 5 – 36
- [100] P. Syverson, "A taxonomy of replay attacks", In Computer Security Foundations Workshop VII. IEEE Computer Society Press, 1994
- [101] R. J. Sutton, "Secure Communications: Applications and Management", John Wiley & Sons, Inc. London, 2002, pp. 267 – 269
- [102] W. Wang, and B. Bhargava, "Visualization of Wormholes in Sensor Networks" In Proceedings of ACM Workshop on Wireless Security (WiSe), in conjunction with MobiCom'04, October 2004
- [103] F. Stajano, "Security for Ubiquitous Computing", John Wiley & Sons, Inc. London, 2002, pp. 85 – 105
- [104] B. Schneier, "Applied Cryptography: Protocols, Algorithms and Source Code in C", 2nd edition, John Wiley & Sons, Inc. London, pp. 26 – 27
- [105] Y-C Hu, A. Perrig, D. B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols", Proceeding of the 2003 ACM workshop on Wireless security, ACM Press New York, 2003, pp. 30 – 40
- [106] J. Lunberg, "Routing Security in Ad Hoc Networks", Tech. Rep. Tik110. 501, Helsinki University of Technology, 2000
- [107] P. Michiardi and R. Molva, "Ad hoc network security", ST Journal of System Research, Volume 4, N1, March 2003
- [108] I. Aad, J-P Hubaux, E. W. Knightly, "Denial of Service Resilience in Ad Hoc Networks", in The 10th ACM International Conference on Mobile Computing and Networking, MobiCom 2004
- [109] B. Awerbuch, R. Curtmola, D. Holmer, C. Nita-Rotaru, H. Rubens, "Mitigating Byzantine Attacks in Ad Hoc Wireless Networks", Technical Report Version 1, Department of Computer Science, Johns Hopkins University & Purdue University, <http://citeseer.ist.psu.edu/661767.html>
- [110] A. Kerckhoffs, "La cryptographie militaire (military cryptography)", Journal de Science militaires, vol. 9, 1883, pp. 5 – 38, <http://www.cl.cam.ac.uk/~fapp2/kerckhoffs>
- [111] F. A. P. Petitcolas, R. J. Anderson, Markus G. Kuhn, "Information Hiding - A Survey", Proceedings of the IEEE, special issue on protection of multimedia content, vol. 87, no. 7, 1999, pp. 1062 – 1078
- [112] P. Wayner, "Disappearing Cryptography: Information Hiding - Steganography and Watermarking", 2nd Edition, Morgan Kaufmann Publishers, London, 2002
- [113] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, "Handbook of Applied Cryptography", CRC Press, London, 1997, pp. 11 – 45
- [114] R. B. Lee, R. L. Rivest, M. J. B. Robshaw, Z. J. Shi, and Y. L. Yin, "Permutation operations in cipher design", in Proc. Intl. Conf. on Information Technology (ITCC)s, Apr. 2004

- [115] B. Preneel, V. Rijmen, and A. Bosselaers, "Recent Developments in the Design of Conventional Cryptographic Algorithms," Proc. Computer Security and Industrial Cryptography, Lecture Notes in Computer Science, Vol. 1528, Springer-Verlag, New York, 1998, pp. 106-131
- [116] N. Ferguson, B. Schneier, "Practical Cryptography", John Wiley & Sons, Inc. London, 2003, pp. 43 – 82
- [117] National Institute of Standards and Technology, "Data Encryption Standard (DES)", December 30, 1993, FIPS PUB 46-2, pp. 51 – 53, <http://www.itl.nist.gov/fipspubs>
- [118] National Institute of Standards and Technology, "Data Encryption Standard (DES)", 1999, FIPS PUB 46-3, pp. 51 – 52, <http://www.csrc.ncsl.nist.gov/fips>
- [119] R. Anderson, E. Biham, L. Knudsen, "Serpent: A proposal for the Advanced Encryption Standard", In National Institute of Standards and Technology, AES Round 1 Technical Evaluation, CD-1 Documentation, 1998, <http://www.cl.cam.ac.uk/~rja14/serpent.html>
- [120] B. Schneier, J. Kelsey, D. Wagner, C. Hall, N. Ferguson, D. Whiting, "The twofish encryption algorithm : a 128-bit block cipher", John Wiley & Sons, Inc. London, 1999
- [121] J. Daemen, V. Rijmen, "Design of Rijndael: Aes - the Advanced Encryption Standard (Information Security & Cryptography S.)", Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 2002
- [122] I Goldberg, An analysis of the Wired Equivalent Privacy protocol, Black Hat briefings, University of Berkeley, <http://www.cypherpunks.ca/bh2001/mgp00001.html>, 2001
- [123] J R Walker, IEEE P802.11 Wireless LANs Unsafe at any key size: An analysis of the WEP encapsulation, IEEE 802.11 Committee 802.11-00/362, 2000.
- [124] N Borisov, I Goldberg and D Wagner, "Security of the WEP algorithm", Department of Computer Science, University of Berkeley, <http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html>, 2001
- [125] N. Borisov, I. Goldberg and D. Wagner, Intercepting mobile communications: the insecurity of 802.11, Proc. of the 7th Annual International Conference on Mobile Computing and Networking, July 16-21, 2001
- [126] H. X. Mel, D. Baker, "Cryptography Decrypted", Addison Wesley, London, 2001, pp. 63 – 75
- [127] W. Diffie, M. E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. IT-22, no. 6, November 1976, pp. 644 – 654
- [128] U. Maurer, S. Wolf, "The Diffie-Hellman protocol. Designs, Codes, and Cryptography 19", KluwerAcademic Publishers, 2000, pp. 147 – 171
- [129] U. M. Maurer and S. Wolf, "On the complexity of breaking the Diffie-Hellman protocol", Tech. Rep. 244, Computer Science Department, ETH Zurich, April 1996
- [130] U. Maurer and S. Wolf, "Diffie-hellman oracles", In Proc. Of CRYPTO 96, 1996, pp. 268 – 282

- [131] M. Steiner, G. Tsudik, M. Waidner, "Diffie-Hellman key distribution extended to group communication", Proceedings 3rd ACM Conference on Computer and Communications Security, 1996, pp. 31 – 37
- [132] R. L. Rivest, Shamir, A. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, vol. 21, no. 2, 1978, pp. 120 – 126
- [133] RSA Laboratories, PKCS#1 v.2.1: RSA Cryptography Standard, January 2001, <http://www.rsasecurity.com/rsalabs/pkcs>
- [134] B. Schneier, "Applied Cryptography: Protocols, Algorithms and Source Code in C", 2nd edition, John Wiley & Sons, Inc. London, pp. 210 – 213
- [135] D. Naccache, D. M. Raihi, S. Vaudenay, D. Rphaeli, "Can D.S.A. be Improved? Complexity Trade-Offs with the Digital Signature Standard", in A. De Santis, Ed., Advances in Cryptology, EUROCRYPT '94, vol. 950 of Lecture Notes in Computer Science, pp. 77 – 85
- [136] U.S. Department of Commerce, National Institute of Standards and Technology, "Digital Signature Standard", Federal Information Processing Standard (FIPS) Publication 186, 1994
- [137] R. Merkle, "One-way hash functions and DES", Crypto 89, LNCS 435, pp. 428 – 446
- [138] M. Wegman and L. Carter, "New hash functions and their use in authentication and set equality", Journal of Comp. and System Sciences, vol. 22, 1981, pp. 265 – 279
- [139] M. Bellare, R. Canetti and H. Krawczyk, "Keying hash functions for message authentication", Advances in Cryptology - CRYPTO '96, Lecture Notes in Computer Science, vol. 1109, Springer-Verlag, 1996, pp. 1 — 15
- [140] L. Carter, M. Wegman, "Universal hash functions, Journal of computer and system sciences", vol. 18, 1979, pp. 143 – 154
- [141] R. Canetti, D. Micciancio and O. Reingold, "Perfectly One-Way Probabilistic Hash Functions", Proceedings of 30th STOC, 1998
- [142] M. Naor, M. Yung, "Universal One-way Hash Functions and their Cryptographic Applications", 21st Symposium on Theory of Computing (STOC), 1989, ACM, New York, pp. 33 – 43
- [143] T. Johansson, "Bucket hashing with small key size, Advances in Cryptology" EUROCRYPT '97, Lecture Notes in Computer Science, Springer-Verlag, 1997
- [144] R. Rivest, "The MD5 message-digest algorithm", Internet Engineering Task Force (IETF), RFC 1321, April 1992
- [145] National Institute of Standards and Technology, "Secure Hash Standard", April 17, 1995, FIPS PUB 180-1, pp. 88 – 89, <http://www.itl.nist.gov/fipspubs>
- [146] R.L. Rivest, "The MD4 message digest algorithm", In S. Vanstone, editor, Advances in Cryptology - CRYPTO'90, LNCS 537, Springer Verlag, 1991, pp. 303 – 311
- [147] National Institute of Standards and Technology, "Secure Hash Standard (draft)", 2001, FIPS PUB 180-2, pp. 89 – 90, <http://csrc.nist.gov/encryption/shs/dfips-180-2.pdf>

- [148] M. Fisher, N. Lynch, M. Paterson, "Impossibility of distributed consensus with one faulty process", *Journal of ACM*, vol. 32, 1985, pp. 374 – 382
- [149] P. Papadimitratos, Z.J. Haas, "Securing the Internet routing infrastructure", *IEEE Communications Magazine*, vol. 40, no. 10, 2002, pp. 60 – 68
- [150] D. E. Denning, "An Intrusion-detection Model", *IEEE Transactions in Eng.*, vol. SE-13, no. 2, Feb. 1987, pp. 222 – 232
- [151] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network Intrusion Detection", *IEEE Network*, May/June 1994, pp. 226 – 41
- [152] K. E. Siroir, S. T. Kent, "Securing the Nimrod Routing Architecture", *Prac. Symposium on Network and Distributed System Security*, Los Alamitos, CA, Feb. 1997, The Internet Society, IEEE Computer Society Press, pp. 74 – 84
- [153] R. Perlman, "Network layer Protocols with Byzantine Robustness", Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1988
- [154] B. Kumar, "Integration of security in network routing protocols", *SIGSAC Reviews*, vol. 11, no. 2, 1993, pp. 18 – 25
- [155] S. Murphy, J. J. Garcia-Luna-Aceves, "An efficient routing Algorithm for Mobile Wireless Networks", *MONET*, Oct. 1996, vol. 1, no. 2, pp. 183 – 197
- [156] B. R. Smith, S. Murphy, J. J. Garcia-Luna-Aceves, "Securing Distance-Vector Routing Protocols", in *Proc. Of the Symposium on Network and Dist. Sys. Security*, Los Alamitos, CA, Feb. 1997, pp. 85 – 92
- [157] S. Jacobs, M. S. Corson, "MANET Authentication Architecture", Internet draft (draft-jacobs-imep-auth-arch-01.txt), Feb. 1999
- [158] Y. Desmedt, Y. Frankel, "Threshold Cryptosystems", *Proc. Advance in Cryptology CRYPTO'89*, LNCS 435, Springer-Verlag, 1989, pp. 307 – 315
- [159] Y. Desmedt, "Threshold cryptography", *European Transactions on Telecommunications*, vol. 5, no. 4, 1994, pp. 449 – 457
- [160] M. Narasimha, G. Tsudik, H. Y. Jeong, "On the utility of distributed cryptography in P2P and MANETs: the case of membership control", in *proc. of the 11th IEEE International Conference on Network Protocols (ICNP'03)*, 4-7 Nov. 2003, pp. 336 – 345
- [161] M.A. Marsh, F.B. Schneider, "CODEX: a robust and secure secret distribution system", *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, 2004 pp. 34 – 47
- [162] Y. Huang; D. Rine, X. Wang, "A JCA-based implementation framework for threshold cryptography", *Proceedings 17th Annual Applications Conference (ACSAC 2001)*, Dec. 2001, pp. 85 – 91
- [163] M. Burrows, M. Abadi, and R. Needham, "A Logic of Authentication", *ACM Operating System Review*, vol. 23, no. 5, 1989, pp. 1 – 13
- [164] L. Gong, R. Needham, and R. Yahalom, "Reasoning about Belief in Cryptographic Protocols," *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, IEEE Computer Society Press, Los Alamitos, CA, 1990, pp. 234 – 248

- [165] P. Papadimitratos, Z. J. Haas, "Secure Routing for Mobile Ad Hoc Networks", in Proc. of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS), San Antonio, Texas, 2002
- [166] N. Asokan, P. Ginzboorg, "Key Agreement in Ad Hoc Networks", Computer Communications vol. 23, no. 17, 2000, pp. 1627 – 1637
- [167] S. Buchegger, J.Y.L. Boudec, "Cooperative Routing in Mobile Ad hoc Networks: Current Efforts Against Malice and Selfishness", In Lecture Notes on Informatics, Mobile Internet Workshop, Informatik 2002
- [168] L. Hu and D. Evans, "Secure Aggregation for Wireless Networks", Proc. of Workshop on Security and Assurance in Ad hoc Networks, Orlando, Florida, Jan 28, 2003
- [169] A. Perrig, R. Canetti, D. Song, J. D. Tygar, "Efficient and Secure Source Authentication for Multicast", In Network and Distributed System Security Symposium, (NDSS'01), February 2001, pp. 35 – 46
- [170] A. Perrig, R. Canetti, J.D. Tygar, D. Song, "Efficient Authentication and Signing of Multicast Streams over Lossy Channels", In IEEE Symposium on Security and Privacy, May 2000, pp. 56 – 73
- [171] P. Bak, "How Nature Works: The Science of Self-Organised Criticality", Copernicus Press, New York, 1996
- [172] D. M. Raup, "Extinctions: Bad Genes or Bad Luck?", Oxford University Press, 1993
- [173] D. M. Raup, "A kill curve for Phanerozoic marine species", Paleobiology vol. 17, 1991, pp. 37 – 48
- [174] B. B. Mandelbrot, "The variation of certain speculative prices", Journal of Business, University of Chicago, vol. 36, 1963, pp. 307 – 317
- [175] B. Gutenberg, C. F. Richter, "Seismicity of the Earth", Princeton University Press, University of Princeton, 1949
- [176] P. Bak, K. Sneppen, "Punctuated equilibrium and criticality in a simple model of evolution", Physics Review Letters, vol. 71, 1993, pp. 4083 – 4086
- [177] K. Sneppen, P. Bak, H. Flyvbjerg, M.H. Jensen, "Evolution as a self-organized critical phenomenon", Proc. National Academy of Science, vol. 92, vol. 1995, pp. 5209 – 5213
- [178] P. Bak, C. Tang, K. Wiesenfeld, "Self-organized criticality: an explanation of 1/f-noise", Physics Review Letters, vol. 59, 1987, pp. 381 – 384
- [179] C. Specjalski, "Sandpile Ver. 1.1", Institute for theoretical physics, Westfälische Wilhelms-Universität Münster, Münster, Germany, <http://pauli.uni-muenster.de/~special/java/sandpile/ver1.1/>, 1998
- [180] H. Haken, "Synergetics", Springer – Verlag, Berlin, 1983
- [181] G. Nicolis, I. Prigogine, "Self-Organisation in Non-Equilibrium Systems", John Wiley & Sons, New York, 1977
- [182] J.-L. Deneubourg, S. Goss, N. Franks, J. M. Pasteels, "The Blind leading the Blind: Modelling Chemically Mediated Army Ant Raid Patterns", Journal of Insect Behavior, vol. 2, 1989, pp. 719 – 725
- [183] G. W. Flake, "The Computational Beauty of Nature, Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation", Cambridge, MA: MIT Press, 1998, pp. 4 – 5

- [184] C. H. Bennett, "How to Define Complexity in Physics, and Why", In: Complexity, Entropy and the Physics of Information. (Ed: Zurek,WH) Addison-Wesley, Redwood City California, 1990, pp. 137 – 148
- [185] J.L. Casti, "Connectivity, Complexity and Catastrophe", John Wiley, NY, 1979
- [186] P. Grassberger, "Problems in Quantifying Self-organized complexity", Helvetica Physica Acta vol. 62, 1989, pp. 498 – 511
- [187] J. M. Seaman, F. Koenig, "A Comparison Measures of Cognitive Complexity", Sociometry, vol. 37, 1974, pp. 375 – 390
- [188] W. B. Rouse, S. H. Rouse, "Measures of Complexity of Fault Diagnosis Tasks", IEEE Trans. Syst. Man Cybernetics, vol. 9, 1979, pp. 720 – 727
- [189] R. Wackerbauer, A. Witt, H. Altmanspracher, J. Kurths, H. Scheingraber, "A Comparative Classification of Complexity Measures based on distinguishing partitions in phase space as well as structural v. dynamic elements", Chaos, Solitons and Fractals vol. 4, 1994, pp. 133 – 173
- [190] B. Edmonds. "Syntactic Measures of Complexity, PhD thesis", University of Manchester, <http://www.cpm.mmu.ac.uk/~bruce/thesis>, 1999
- [191] A. N. Kolmogorov, "Three approaches to the quantitative definition of information", Problems Information Transmission, vol. 1, 1965, pp. 1 – 7
- [192] G. J. Chaitin, "On the length of programs for computing finite binary sequences", J. Assoc. Comput. Machines, vol. 13, 1966, pp. 547 – 569
- [193] C. Emmeche, "The Garden in the Machine", University of Princeton, Princeton, 1994
- [194] M. Li, P. Vitanyi, "An Introduction to Kolmogorov Complexity and its Applications", Springer, New York, 2nd edition, 1997
- [195] R. K. Standish "On complexity and emergence", Los Alamos Physics Archive arXiv:nlin.AO/0101006, <http://xxx.lanl.gov/abs/nlin/0101006>, 2001
- [196] M. Schroeder, "Fractals, Chaos, Power Laws", Freeman Press, New York, 1991
- [197] M. Gell-Mann. "The Quark and the Jaguar: Adventures in the Simple and the Complex", Freeman Press, London, 1994
- [198] J.D. Lohn, W.F. Kraus, D.S. Linden, "Evolutionary Optimization of a Quadrifilar Helical Antenna", Proc. of the IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting, Vol. 3, Jun. 2002, pp. 814 – 817
- [199] M. Resnick, "Turtles, Termites, and Traffic Jams, Explorations in Massively Parallel Microworlds", Cambridge, MA: MIT Press, 1994
- [200] J. Holland, "Emergence. From Chaos to Order", Cambridge, MA: Perseus Books, 1998
- [201] A. Koestler, "The ghost in the machine", Hutchinson & Co, London, 1967
- [202] G. Beni, "The Concept of Cellular Robotic System", In Proceedings of the IEEE Int. Symp. On Intelligent Control, Los Alamitos, CA, IEEE Computer Society Press, 1988, pp. 57 – 62

- [203] G. Beni, J. Wang, "Theoretical Problems for the Realization of Distributed Robotic Systems", In Proc. of the IEEE International Conference on Robotic and Automation, Los Alamitos, CA, IEEE Computer Society Press, 1991, pp. 1914 – 1920
- [204] S. Hackwood, G. Beni, "Self-Organising Sensors by Deterministic Annealing", In Proc. of the IEEE/RSJ International Conference on Intelligent Robot and Systems, IROS'91, Los Alamitos, CA, IEEE Computer Society Press, 1991, pp. 1177 – 1183
- [205] S. Hackwood, G. Beni, "Self-Organisation of Sensors for Swarm Intelligence", In Proc. of the IEEE International Conference on Robotics and Automation, Los Alamitos, CA, IEEE Computer Society Press, 1992, pp. 819 – 829
- [206] E. Bonabeau, M. Dorigo, G. Théraulaz, "Swarm Intelligence: From Natural to Artificial Systems", Oxford University Press, Oxford, 1998
- [207] P. P. Grasse, "La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. La theorie de la stigmergie: essai d'interpretation du comportement des termites constructeurs", Insectes Sociaux, vol. 6, 1959, pp. 41 – 81
- [208] G. Theraulaz, E. Bonabeau, "A brief history of stigmergy", Artificial Life vol. 5, 1999, pp. 97 – 116
- [209] N. Baas, "Emergence, Hierarchies and Hyperstructures" Artificial Life III, MIT Press, 1992, pp. 515 – 537
- [210] D. Gross, B. McMullin, "Is It the Right Ansatz? ", Artificial Life VII, MIT Press, 2001, pp. 355 – 365
- [211] S. Rasmussen, N. Baas, "Ansatz for Dynamical Structures", Artificial Life VII, MIT Press, 2001
- [212] M. A. Bedeau, "Weak Emergence", Philosophical Perspectives, vol. 11, 1997, pp. 375 – 399
- [213] M. A. Bedau, "Weak Emergence." In James Tomberlin, ed., Philosophical Perspectives: Mind, Causation, and World, vol. 11, Blackwell Publishers, 1997
- [214] E. Bonabeau, "Predicting the Unpredictable", Harvard Business Review, vol. 80 no. 3, 2002, p109 – 117
- [215] O. Selfridge, "Pandemonium: A paradigm for learning", in symposium of the mechanisation of thought process, HM Stationery office, London, 1959
- [216] P. Lindsay, D. A. Norman, "Human Information Processing", Freeman Press, San Francisco, 1972, pp. 115 – 117
- [217] D. A. Medler, "A Brief History of Connectionism", Biological Computation Project, Department of Psychology, Alberta University, http://neuron-ai.tuke.sk/NCS/VOL1/P3_html/vol1_3.html, Canada, 1998
- [218] J. Martin "Survival of the Institutionally Fittest Concepts", Journal of Memetics: Evolutionary Models of Information Transmission, vol. 3, 1998, http://www.cpm.mmu.ac.uk/jom-emit/1999/vol3/de_jong_m.html
- [219] J. Roughgarden , A. Bergman, S. Shafir, C. Taylor, "Adaptive Computation in Ecology and Evolution: A Guide for Future Research", In R.K. Belew and M. Mitchell (eds.) Adaptive Individuals in Evolving Populations: Models and Algorithms, Santa Fe Institute Studies in the Sciences of Complexity. Addison Wesley, Reading, MA, 1996

- [220] J. H. Holland, "Adaptation in Natural and Artificial Systems", MIT Press, 1992, pp. 20 – 31
- [221] S. Johnson, "Emergence: The connected lives of Ants, Brains, Cities and Software", Penguin Press, 2001, pp. 163 – 189
- [222] W Beebe, "Edge of the Jungle", Henry Holt and Company, New York, 1921
- [223] Various, "OPNET Modeller", OPNET Technologies Inc.
<http://www.opnet.com>
- [224] S. McCanne, S. Floyd, "ns-Network Simulator", <http://www-mash.cs.berkeley.edu/ns>
- [225] The CMU Monarch projet, "The cmu monarch project's wireless and mobility extension to ns", <http://www.monarch.cs.cmu.edu>
- [226] M. Resnick, "StarLogo: An environment for decentralized modeling and decentralized thinking", CHI'96 Conference Companion, (Vancouver, Canada, April 13 -18), 1996, pp. 11 – 12
- [227] Vanessa Stevens Colella, Eric Klopfer, Mitchel Resnick, "Adventures in Modeling: Exploring Complex, Dynamic Systems with StarLogo", Teachers College Press, London, 2001, pp. 11 - 12
- [228] M.M. Zonoozi, P. Dassanayake, "User mobility modeling and characterization of mobility patterns", IEEE Journal on Selected Areas in Communications 15, no. 7, 1997, pp. 1239 – 1252
- [229] G. Holland, N. H. Vaidya, "Analysis of TCP Performance Over Mobile Ad Hoc Networks", Proceedings of IEEE/ACM MOBICOM '99, 1999, pp. 219 – 230
- [230] R. Bouchard, C. Pyers, "Use of gravity model for describing urban travel", US Highway research record, 88, 1965, pp. 1 – 43
- [231] X. Hond, M. Gerla, G. Pei, C. Chiang, "A group mobility model for ad hoc wireless networks", In proc. Of the ACM International Workshop on Modelling and Simulation of Wireless and Mobile Systems, 1999, pp. 53 – 60
- [232] K. H. Wang, B. Li, Group Mobility and Partition Prediction in Wireless Ad-Hoc Networks, In Proc. of IEEE International Conference on Communications (ICC), New York, 2002
- [233] J. B. Tracy Camp, Vanessa Davies, "A survey of mobility models for ad hoc network research, Wireless Communications and Mobile Computing", vol. 2, no. 5, 2002, pp. 483 – 502
- [234] B-J Kwak, N-O Song, L. E. Miller, "A mobility measure for mobile ad hoc networks", IEEE Communications Letters, vol. 7 no. 8, 2003, pp. 379 – 381
- [235] Elizabeth M. Royer, Charles E. Perkins, "Transmission range effects on aodv multicast communication", Swedish Workshop on wireless Ad-Hoc Networks, March 2001
- [236] T.D. Dyer, R.V. Boppana, "A comparison of tcp performance over three routing protocols for mobile ad hoc networks", ACM Mobihoc 2001, October 2001
- [237] H . Gomaa, "Designing Concurrent, Distributed, and Real-Time Applications with UML", Addison-Wesley, 2000

- [238] M. Richters, M. Gogolla "On formalizing the UML Object Constraint Language OCL", In T-W Ling, S. Ram, and M. Li Lee, editors, Proc. 17th Int. Conf. Conceptual Modeling (ER'98), Lecture Notes in Computer Science, Springer-Verlag, no. 1507, 1998
- [239] J. B. Warmer, A. G. Kleppe, "The Object Constraint Language: Precise Modeling With UML", Addison-Wesley, 1998
- [240] M. Fowler, K. Scott UML Distilled: Applying the Standard Object Modeling Language. Addison-Wesley, 1997, pp. 99 – 105
- [241] P. Stevens, R. Pooley, "Using UML – Software Engineering with Objects and Components", Addison-Wesley – An imprint of Pearson Education Limited, updated edition, 2000
- [242] J. Cheesman, J. Daniels, "UML Components – A Simple Process for Specifying Component-Based Software", Addison-Wesley, 2000
- [243] Object Management Group (OMG), "OMG Unified Modeling Language Specification", Version 2.0, 2003, <http://www.omg.org/uml/>
- [244] P. Papadimitratos, Z.J. Haas, "Securing the Internet routing infrastructure", IEEE Communications Magazine, vol. 40, no. 10, 2002, pp. 60 – 68
- [245] D. E. Denning, "An Intrusion-detection Model", IEEE Transactions in Eng., vol. SE-13, no. 2, Feb. 1987, pp. 222 – 232
- [246] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network Intrusion Detection", IEEE Network, May/June 1994, pp. 226 – 41
- [247] K. A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, R. A. Olsson, "Detecting Disruptive Routers: A Distributed Network Monitoring Approach", in Proc. of the IEEE Symposium on Research in Security and Privacy, May 1998, pp. 115 – 124
- [248] S. Cheung, K. Levitt, "Protecting Routing Infrastructures from Denial of Service Using Cooperative Intrusion Detection", In The 1997 New Security Paradigms Workshop, September 1998, pp. 94 – 106
- [249] H. M. Deitel, P. J. Deitel, "Java – How to Program", Prentice Hall, 3rd ed., 1999, pp. 144 – 145
- [250] B. R. Smith, S. Murthy, J. J. Garcia-Luna-Aceves, "Securing Distance Vector Routing Protocols", In Symposium on Network and Distributed Systems Security (NDSS'97), February 1997